HammerBots • Aqua Bot • What Is A Bot?

SERVO MAGAZINE

FOR THE ROBOT INNOVATOR

www.servomagazine.com

July 2010

• TANKBOT •
Beginner's Guide And Future Plans

Gulf Oil Spill Disaster

Meet The ROVs that tried to save the day

✦ Real-Time GPS Tracking System
• GPS Module
• Local Web Server
• Simple Web Interface
• Google Maps API

✦ Coding An Encoder
Adding An Optical Encoder As An Input Device

✦ RoboGames In Retrospect
Insights and Lessons Learned At RoboGames 2010

U.S. $5.50 CANADA $7.00

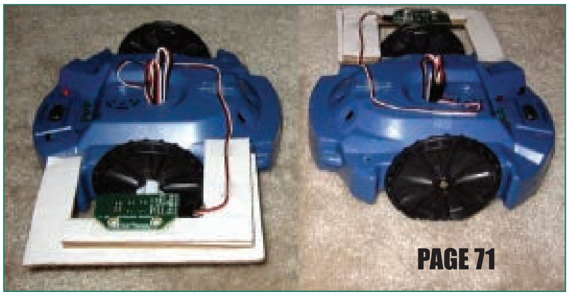
07

0 71486 02422 4

SERVO MAGAZINE

07.2010

VOL. 8 NO. 7



PAGE 71

PAGE 10

Columns

Departments

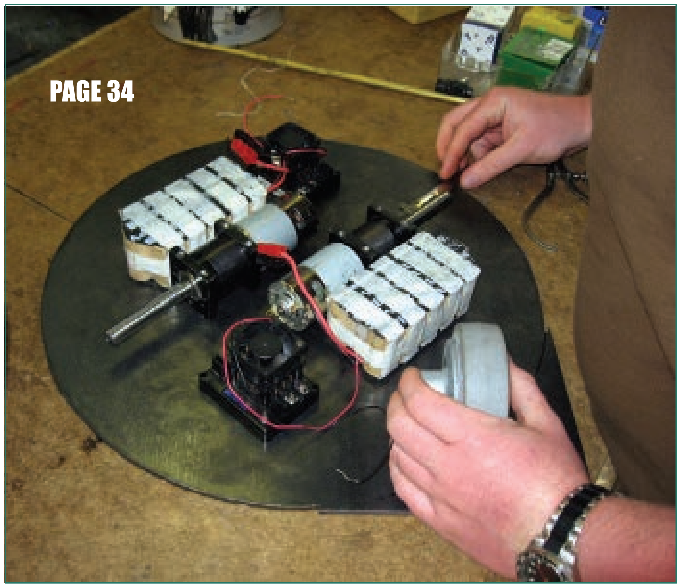
The Combat Zone...

Features

Events

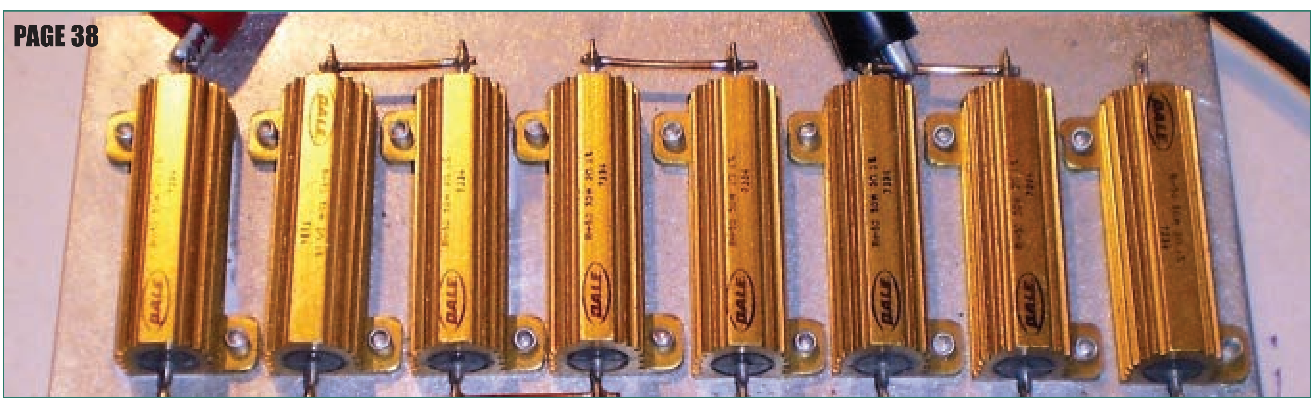
SERVO Magazine **(ISSN 1546-0592/CDN Pub Agree#40702530)** is published monthly for $24.95 per year by T & L Publications, Inc., 430 Princeland Court, Corona, CA 92879. PERIODICALS POSTAGE PAID AT CORONA, CA AND AT ADDITIONAL ENTRY MAILING OFFICES. POSTMASTER: Send address changes to **SERVO Magazine, P.O. Box 15277, North Hollywood, CA 91615** or Station A, P.O. Box 54, Windsor ON N9A 6J5; cpcreturns@servomagazine.com

In This Issue ...



PAGE 34

PAGE 43



PAGE 38

Mind / Iron

by Bryan Bergeron, Editor

Robotic Chia Pets

As a young experimenter, I remember trying to use the ground return current from the city power station to power something — anything. By driving copper stakes in the ground about two meters apart, I was able to create a 500 mV AC source. At the time, it wasn't enough to power any electronic device I had on hand. Today — thanks to developments in energy harvesting, including devices designed to work at very low voltages — 500 mV is huge. The only problem is, it's AC and not DC.

There are natural sources of DC voltage that can and have been harvested, including plants. It's been documented since at least the 1970s that plants generate a potential between the plant tissue and the soil. More recently, with the advent of ultra-low-power devices, experimenters have used plant 'batteries' together with charge pumps to power digital counters and other electronic circuitry.

For example, third place in the 2010 International Design MSP430 Ultra-Low Power Challenge (sponsored by Texas Instruments) was a device run by a plant battery. The experimenter, Carlos Cossio, connected five potted plants in series to achieve a voltage of 3.2V open circuit — enough to power the MSP430 and some associated hardware.

The voltage generated by a plant is due to the pH gradient in its tissues, as opposed to some sort of chemical reaction. In other words, this isn't one of those lemon juice as an electrolyte battery, but a different kind of battery based on a different mechanism. Because the potential is between the soil and plant tissue, you have to use separate pots if you want to connect the plants in series to create a higher voltage.

As you might expect, plant batteries have a high impedance and relatively low voltage, and the voltage varies from one plant species to the next. The common jade plant is an excellent voltage source, in case you'd care to experiment on your own. For contacts, you need a nail or other conductor stuck in the potted soil, and a straight pin/alligator clip to get at the tissue of the plant. I imagine that a long-term battery connection needs to be sealed from microbes that could destroy the plant tissue.

From the title of this editorial, you can probably guess where I'm going with this ... plant-powered robots. I'm talking about small, plant-powered motors, sensors, and control electronics. There's a body of literature out there on BEAM robotics that suggests this is doable. I don't envision jade-covered mechatronics circling underfoot, but perhaps a small carpet roamer powered by hundreds of smaller plants. You'll have to design the system so that daily watering doesn't destroy the onboard electronics and motors.

Before you start building your motorized chia pet robot, I suggest you read the excellent article, *Ultra-Low Voltage Nanoelectronics Powered Directly, and Solely, from a Tree*, by Carlton Himes, Eric Carlson, and others, published in the *IEEE Transactions on Nanotechnology*, 2009. There are full reprints on the Web.

By the way, in case you're thinking about ways of cashing in on this area, there's at least one patent that has been issued by the US patent office on plant batteries. Check the patent office (**www.uspto.gov**) for details. You can also check our sister publication, *Nuts & Volts*, for an upcoming article by Carlos Cossio on his entry in the Texas Instruments contest.

If you've seen *The Matrix*, you know that animals also generate electrical potentials. There may come a day when we each have communicators or at least mp3 players sewn into our skin, with electrodes taking power from our muscle tissue. Perhaps small rodents will provide electrical power as well as cortically-directed control of robots. However, plants are a better bet in the near-term — they're more politically correct. SV

FOR THE ROBOT INNOVATOR
SERVO MAGAZINE

Published Monthly By
T & L Publications, Inc.
430 Princeland Ct., Corona, CA 92879-1300
(951) 371-8497
FAX **(951) 371-3052**
Webstore Only **1-800-783-4624**
www.servomagazine.com

Subscriptions
Toll Free **1-877-525-2539**
Outside US **1-818-487-4545**
P.O. Box 15277, N. Hollywood, CA 91615

PUBLISHER
Larry Lemieux
publisher@servomagazine.com

ASSOCIATE PUBLISHER/ VP OF SALES/MARKETING
Robin Lemieux
display@servomagazine.com

EDITOR
Bryan Bergeron
techedit-servo@yahoo.com

CONTRIBUTING EDITORS
Jeff Eckert
Jenn Eckert
Tom Carroll
David Geer
Dennis Clark
R. Steven Rainwater
Fred Eady
Kevin Berry
Dave Prochnow
Clark Robbins
Forrest Stanley
Gregory Intermaggio
Thomas Kenney
Pete Smith
Bryce Woolley
Evan Woolley
Calvin Turzillo

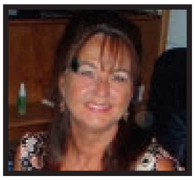
CIRCULATION DIRECTOR
Tracy Kerley
subscribe@servomagazine.com

MARKETING COORDINATOR WEBSTORE
Brian Kirkpatrick
sales@servomagazine.com

WEB CONTENT
Michael Kaudze
website@servomagazine.com

ADMINISTRATIVE ASSISTANT
Debbie Stauffacher

PRODUCTION/GRAPHICS
Shannon Christensen



Robytes

by Jeff and Jenn Eckert

ROVs Aid BP

The Maxximum ROV from Oceaneering International.

It has been widely reported that robotic submarines were deployed within hours of the BP oil disaster, but details were sketchy. Were they Navy subs? Specially modified UAVs? Well, it turns out that Oceaneering International, Inc. (**www.oceaneering.com**), has been offering a variety of products and services geared toward offshore oil exploration and production for years. Included is a line of remotely operated vehicles, eight of which were attempting to awaken the well's blowout valve as of this writing. And we're not talking about dainty little tadpoles.

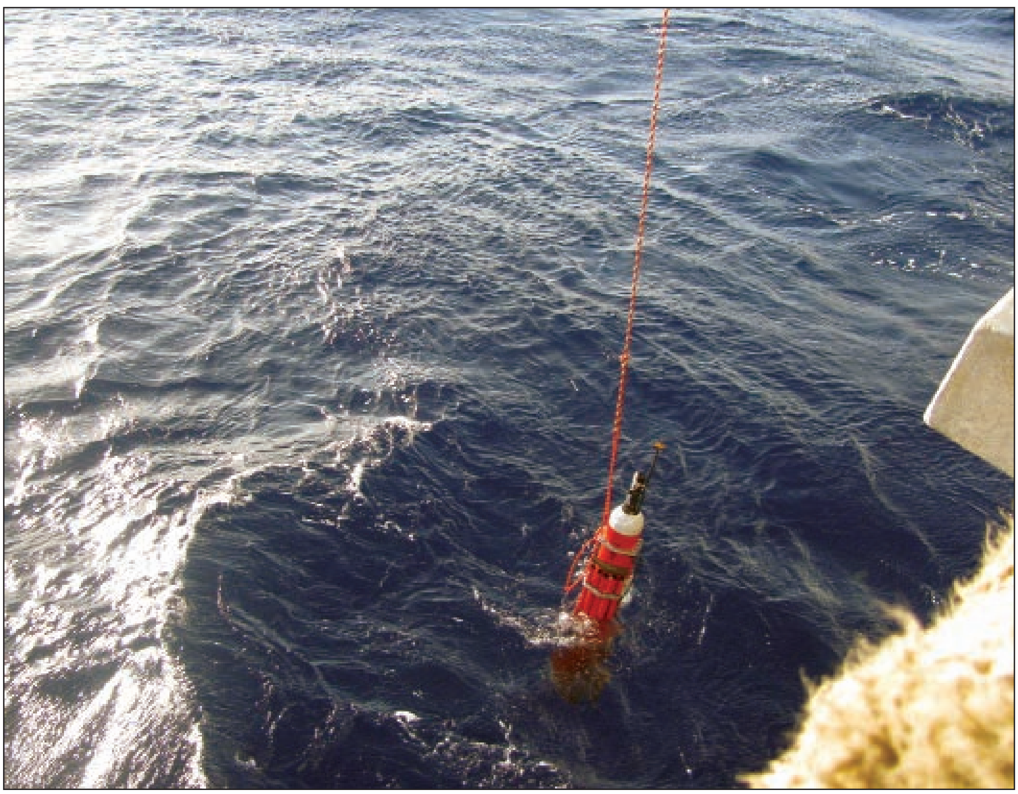
Oceaneering's Maxximum® model is a dual-manipulator, 300 HP slogger that — with its enhanced thruster configuration — offers nearly a ton and a half of center lift capacity and forward pivoted bollard lift greater than 1,800 lb. It also sports microprocessor-based telemetry and a direct fiber optic link to the console, providing a transmission path for video (eight channels) and data signals. The standard unit is rated for depths of 10,000 ft. Weighing in at 10,750 lb in air, it can carry a payload of up to 1,100 lb. For more indepth coverage on the ROVs used in this effort, check out the article on page 55 of this issue.

UAV Achieves Perpetual Motion (Almost)

Okay, it's not perpetual motion in an absolute sense, as it actually does suck a little energy out of the environment. But the Sounding Oceanographic Lagrangrian Observer Thermal RECharging (SOLO-TREC) autonomous underwater vehicle uses a clever thermal recharging engine that draws its energy from temperature differences found at different ocean depths.

In a joint effort by NASA's Jet Propulsion Lab (**jpl.nasa.gov**) and the Scripps Institution of Oceanography (**scripps.ucsd.edu**), the UAV has already completed more than 300 dives to depths of up to 500 m (1,640 ft). The secret is a set of 10 external tubes that are filled with a waxy phase-change material that alternately expands and melts in warm water, and solidifies and shrinks in cold. This puts pressure on an oil reservoir that drives a hydraulic motor, and the motor produces juice to recharge the sub's batteries. You get about 1.7 watt-hours (6,100 joules) per dive which is enough to run the vehicle's instruments, communications equipment, a GPS unit, and the buoyancy-control pump. No, it doesn't produce enough to provide horizontal propulsion, but what do you want for nothing? As you read this, SOLO-TREC has been redeployed on an extended mission that is planned to last months, if not years.

The SOLO-TREC AUV powered by temperature differentials.



Not-So-Mysterious Robot Plane

The USAF X37B, launched back in April.

Some of the press went into a tizzy over the Air Force's April 22nd launch of the "mysterious" and "secret" X37B space plane, with some even claiming that it's a prototype space weapon of some kind. Apparently, they didn't know that the shuttle-shaped UAV has been around quite a while, with its unpowered X40A predecessor flying seven successful missions in 2001. The concept actually originated with NASA in 1999, but was passed off to DARPA in 2004 because of funding problems. In 2006, the Air Force took over the program and related expenses.

The X37's general specs are readily available, so let us just note that it is designed for space missions of up to 270 days, after which it will fly itself back to Vandenburg AFB. As shown in the artist's rendition, its present mission is to go into orbit, deploy some solar panels, and run its systems through their paces. Built by Boeing's Phantom Works division, the X37B is about 29 ft long with a wingspan of just over 14 ft. It stands just over 9.5 ft tall and weighs about 11,000 lb.This is pretty petite compared to the shuttle's 122 ft length and 78 ft wingspan, but it is described as technologically "one generation beyond the shuttle."

So, when is it going to touch down? "In all honesty, we don't know when it's coming back for sure," noted a spokesman. "It depends on the progress we make with the on-orbit experiments and the on-orbit demonstrations." Well, there you are. A mystery after all! Details on the X37 and practically everything else, are available at **www.af.mil/information/factsheets**.

Voice Development Kit

Meanwhile, back on Earth, if you are working on a bot or virtually anything else that could use voice and speech recognition capabilities, take a look at the new VoiceGP product family from Austria's TIGAL KG (**www.tigal.com**). According to the company, "The VoiceGP family includes all the hardware and software required for easy and cost-effective development and implementation of speech synthesis and multi-language speaker-independent and speaker-dependent speech recognition capabilities to virtually any application."

The product family consists of the VoiceGP module and two development kits with bundled development software. The VoiceGP module is based on Sensory's RSC-4128 mixed signal processor, and it's capable of running the latest Sensory FluentChip ™ core technology libraries. The development kit includes the VoiceGP module and a development board that can be powered via USB, batteries, or an external power supply. The VoiceGP DK, VoiceGP DK-T2SI, and VoiceGP module are priced (respectively) at 149, 299, and 45 Euros (about $194, $389, and $58 this week). **SV**

TIGAL's VoiceGP products allow you to add speech capabilities to your designs.

GEERHEAD

by David Geer
Contact the author at geercom@windstream.net

Aqua Robot

The Aqua Robot — a collaboration of participating universities including McGill University, York University, and Dalhousie University (all in Canada) — swims beneath the water and walks on land either autonomously or tethered to capture environmental images for preservation efforts. While the results and conclusions drawn from a 2005 study delving into the robot's efficacy in aiding the fishing industry remain confidential, many of the robot's known research goals may give us a peek into related applications.

Two different iterations of the Aqua robot with flipper legs for swimming.

Images courtesy of Dudek Lab, McGill University.

There are many water applications. In fact, according to Greg Dudek, director, the School of Computer Science and James McGill Chair, McGill University, "The most ambitious goal is to have the robot replace people for underwater observation." The robot could also provide tools and lighting as it assists people underwater, or collect data and check on human investigators to see that they are getting enough air. Toward these ends, many technologies and capabilities have been and are in the process of being developed or applied, including visual servoing, better cameras, and machine vision for color, motion, and texture recognition, plus trinoculars, better computers and processors, and more.

Technologies

The first Aqua robot (Aqua 1.0) was completed in 2004. The next model (the Aqua 1.5) was completed around 2006-2007 and is named Ramius. The current model — which is technically Aqua 2.0 — is called the Kroy model.

Despite the continual development of newer

The latest Aqua robot with rubber treaded legs for walking. Here, a human operator tethers the robot with a cable for remote communications and manuevers. The researchers have just completed development of a set of six all-purpose legs [not shown] for land and sea so they don't have to change the robot's legs to go into the water from the shore.

models, the older models are still useful and can sometimes be upgraded. "We try to back port new technologies into older models when possible, but that is not always possible," says Dudek.

Speaking of new technologies, the greatest advancement is the new smaller, sleeker body design which allows the robot to operate more quickly. Researchers can also pack it into a single pelican case for easy stowage on planes for flight to remote water destinations.

The computers are also faster and higher performance, as are the cameras. The Kroy has two computers inside and uses three cameras which study the environment and help with navigation. Two of the cameras face forward, the other one is in the back.

The robot's software — RoboDevel — runs on a real-time operating system called QNX. This software generates the different gaits or strides the robot needs to employ on land and under water. It also operates the servo software.

The robot's many gaits include one for very fast strides that are unfortunately not energy efficient, and others are for turning, moving laterally, and so on.

The robot's legs store spring energy when they bend under a load and this energy is released as the legs push forward. This is a critical part of the robot's locomotion strategy. When the robot weighs down on the legs, the compression of the spring stores energy that is released, aiding forward movement.

"The swimming and walking functions are implemented as RHexLib modules. A visual processing module for Aqua is being developed with VXL as the vision library," according to Dudek.

The robot actually runs two operating systems. One of the computers runs Linux and one runs QNX (which is similar to Linux). The computer running QNX moves the legs and tracks their motion. When the robot transitions from water to shore, it is crucial that the timing is accurate when changing gaits. The Linux operating system manages diver-following and long-term planning by the robot.

The robot's programming is very unique in that it takes a lot of research to make a robot swim. "To build a gait is very tricky," says Dudek.

Capabilities

Through the application of visual servoing, the Aqua robot can adjust its course based on what it sees. "It can

Here, the Aqua robot and a diver are working together in a pool. The robot obeys image-based commands delivered by hand in front of its camera. This control technology was developed because the robot cannot receive radio-based commands underwater.

These image-based communications are called RoboChat. All underwater communications must be acoustic or visual. Acoustics are typically too slow. Some of the commands the diver gives the robot are stop, go, follow the diver, surface, etc. The technology also enables the diver to give the robot numerical inputs. The robot can be programmed while under water using these images.

follow a diver or the contours of a reef," explains Dudek. "It learns patterns associated with a particular diver it wants to follow, and looks for those patterns and follows them." These patterns include combinations of colors, motions, and textures.

This is called target recognition. The technology uses a filter that analyzes each picture the robot takes. If it is tracking a face, it will have been trained on what two eyes look like and it will search for those. The technology uses hundreds of filters. This is called ensemble tracking because it uses an ensemble of filters.

In the first stage of training the robot's tracking system, researchers show the robot an image of something they want it to follow. In the second stage, some of the robot's systems train the other systems. The robot's interworkings constantly adapt to what one of its systems knows is a correctly identified target.

In the research at York University, the robot applies trinocular sensors which use two cameras together to see how far apart objects are under water. One camera looks ahead following a diver, while another looks at the terrain below, mapping it out.

Thanks to the computers and cameras, the robot takes better images including videos (watch the lobster video in the resources). When the robot surfaces, researchers may want to know right away if it saw anything of significance. The robot may have taken three hours of video, but with its new technology it can identify the 25 most interesting images very quickly.

The problem with selecting the things that are interesting from all the images is that the technology has to model things that people would find interesting. So, the robot needs to be able to identify those. In the model, some things are interesting because they are new, and some are interesting because they are unlike any previous images.

The Aqua robot works in tandem with stationary sensors in the water. There is a separate project dedicated to deploying these sensors to collect data. An Aqua robot can swim to a sensor, pick up the data, and retrieve it.

The Aqua robot also has legs — a new set that are fully amphibious, so it can walk and swim, and do both very well. Photos are currently unavailable as this technology is still a bit under wraps. Having the Aqua robot swim and then walk directly onto shore when done opens up a lot of possible uses. These include military applications where the bot can stay on the beach and then go into the water to de-mine the surf zone. The robot could also walk into the water, swim and collect data, crawl back out of the water, and walk into a designated area and uplink the data. Or, it can follow a diver into the water.

Final Thought

The RHex-based Aqua robot is blazing a new trail in water research robots. **SV**

Resources

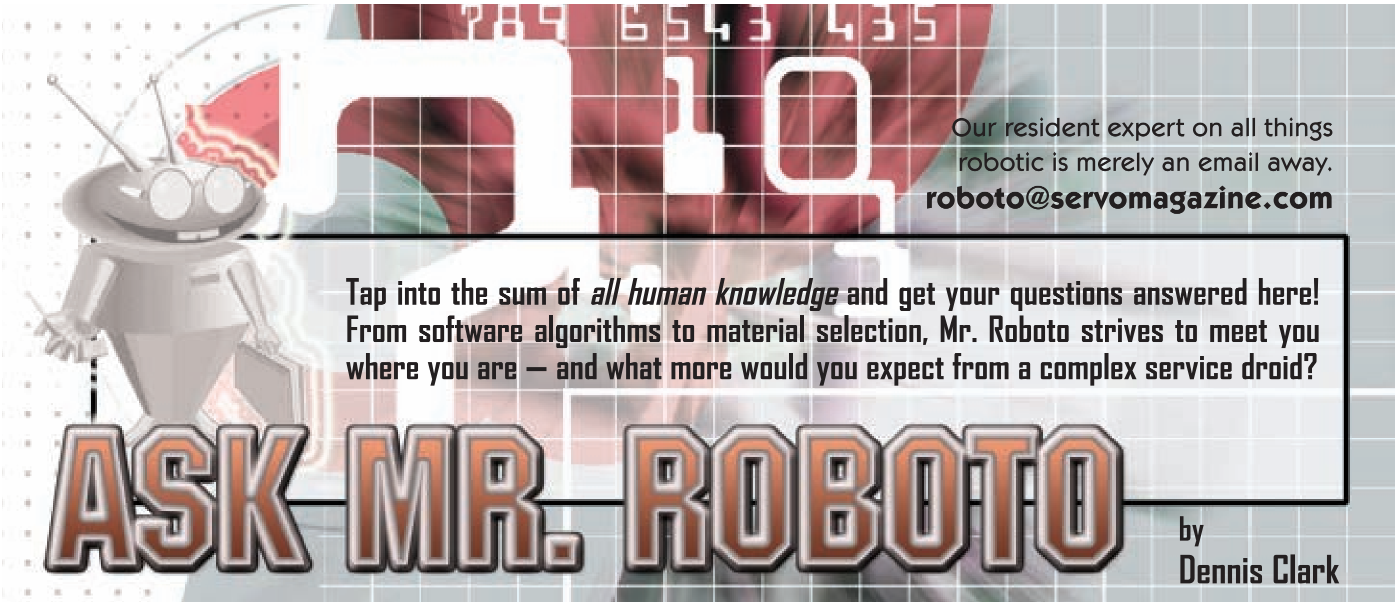
Aqua robot sees lobster (video)
www.youtube.com/watch?v=FCOZFwzMiU8

Aqua robot walks on shore (video)
www.youtube.com/watch?v=pXXikLMXhtY

Aqua robot scales angled sandy terrain (video)
www.youtube.com/watch?v=aiDV06DIqE8

Aqua robot transitions from water to shore with new legs (video)
www.youtube.com/watch?v=lpkXopUHKg8

RHex Robot Platform
www.rhex.web.tr



ASK MR. ROBOTO

Our resident expert on all things robotic is merely an email away.
roboto@servomagazine.com

Tap into the sum of *all human knowledge* and get your questions answered here! From software algorithms to material selection, Mr. Roboto strives to meet you where you are — and what more would you expect from a complex service droid?

by
Dennis Clark

It will be July when you read this, but it is the snowy month of May for me as I write this. So, let's heat things up and get right to the questions.

Q. I have scavenged a Polaroid Spectra sonar transducer and board. I have read your page at **www.seattlerobotics.org/Encoder/200010/dlcsonar.html**. I'm not sure about the purpose of the small board that is above the ranging PCB assembly. What is the function, how does it interact with the sonar unit, and what are the parts? Have you worked with an Arduino board with this application?

— Richard

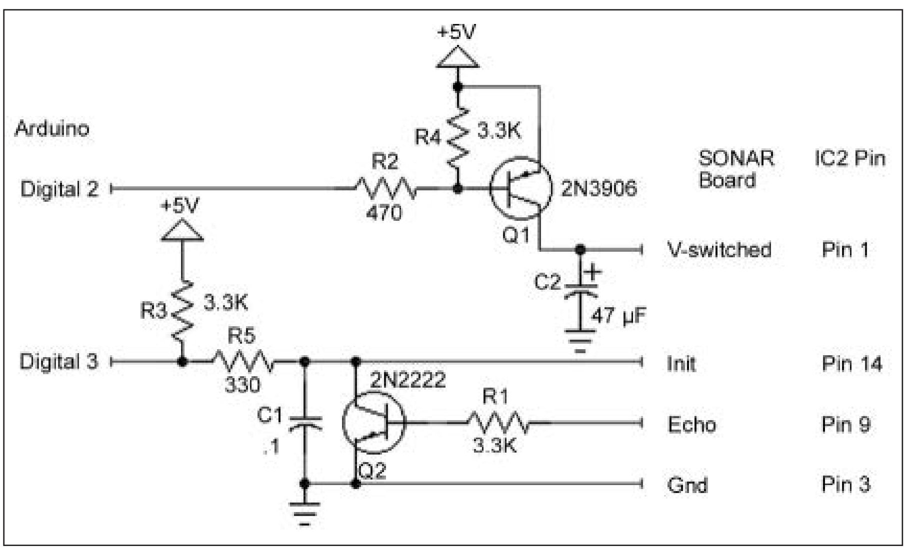
A. Ah, the hacked Polaroid camera SONAR! It doesn't seem as popular as it was 10+ years ago before all of the affordable small SONARs appeared on the market, but we shouldn't ignore it! These SONAR units are not too difficult to get to work and they have a greater range than the commonly used SONARs today. When I wrote that article, it actually came from a web page of mine. I gave example circuits and code for the Parallax BASIC Stamp II and the Motorola 68HC11 — two of the more popular robot controllers of the day. Your question allows me to revisit those glorious hacking days and use the much faster and more convenient Arduino platform. Before your question, I had not tried this on the Arduino. Thanks for getting me to amend that omission! Yes, it does work, but not with the circuit that I showed in my article. The Stamp used a PIC which worked fine with the sub-3V output of that board, the Arduino's AVR processor apparently does NOT like the voltage level of this circuit. I modified the design a little and it works great. **Figure 1** shows my new circuit for using this version of a Polaroid SONAR.

How This Board Works

This circuit is designed to do two things. The first is to programmatically power the SONAR board on and off. The Arduino Digital 2 ping does this via Q1. When D2 is high, the PNP transistor (and SONAR board) is off. When D2 goes low, then Q1 turns on and the SONAR board is ready to work. C2 is a *reservoir* capacitor that stores energy for the high current needs of the SONAR when you *ping* it.

Arduino Digital 3 pin has two lives. The first task D3 must do is hold the *Init* line to the SONAR board low. When you start this program, that is one of the first things done (see **Listing 1**). To tell the SONAR board to issue a *ping*, the program will make D3 an input. This will allow R3 to pull this line high, which starts a SONAR sounding cycle. Immediately after making D3 an input, the program issues a *PulseIn* command which will measure the length of the

Figure 1. SONAR control board schematic.



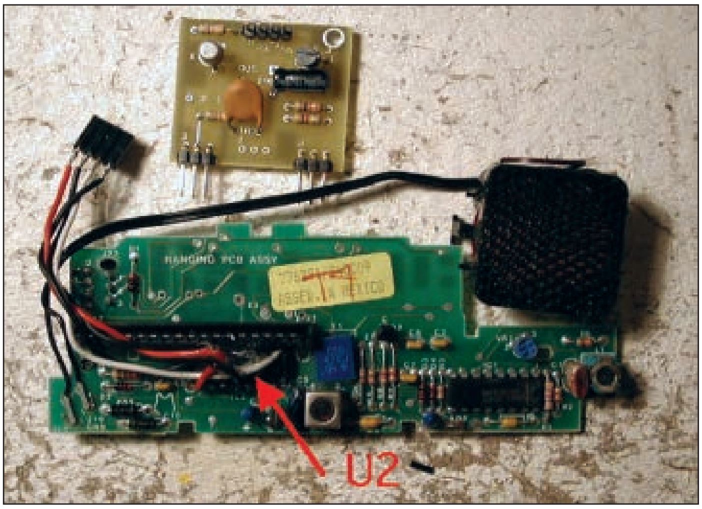


Figure 2. SONAR and control boards.

pulse coming in on D3. The cap C1 delays the rise of the D3 line long enough that the PulseIn command will see a logic low transition to a logic high. This is essential because the PulseIn command will not start timing until it sees the D3 line transition from a low to a high logic level. When the SONAR sees an echo come back, it will drive the *Echo* line high which turns on Q2, which then pulls the D3 line low. This terminates the PulseIn command and returns a timer value in microseconds.

This circuit is a fancy way to handle three I/O lines' work, but uses only two I/O lines because the signal line from the SONAR board does not reach 3V. I added R5 which allows the logic high voltage seen by the Arduino to be over 3V so that the microcontroller will see this as a valid logic level and still have a low resistance path to pull the *Init* line to the SONAR low. The connector on this SONAR board is kind-of odd, so it is easier to solder wires to the SONAR controller chip U2 than it is to make a cable

Listing 1: The Arduino SONAR code.

```
/*
  SONAR
  Uses an old Polaroid SONAR unit to measure
  distance on an Arduino.

  On a cool day (15 C) at sea level the speed
  of sound is about 340.3m/s.
  Because our timer runs in microseconds we'll
  convert that number to cm/us thusly:
  1/34030 * 1,000,000 = about 29.4us/cm.
  Because we're measuring out and back, we
  double that to about 59us/cm. There are
  2.54 inches in a cm, so that number becomes
  2.54 * 29.4 * 2 = about 149us/in.

  The Arduino gives us the pulseIn command which
  will measure the duration of a pulse in
  microseconds. The hacked out Polaroid
  SONARs need to have their power cycled between
  pings, so we'll use an interesting circuit
  that controls the power on the board
  with one I/O pin and will both ping and read
  back the pulse delay with another pin. See
  the accompanying circuit to see how we do
  this. Simple!

  I've found that these boards only work past
  about 18 inches. Some of the Poloroid boards
  will work at ranges as close as 11 inches.
  On the bright side, they work out to over
  10 feet. This is a longer range SONAR than
  your modern day small robot boards.

  Code by Mr. Roboto May 2010
  Use it any way you like.

*/

const int powerPin =  2;    // SONAR board power
const int echoPin = 3;      // Init/Echo line
// The setup() method runs once, when the
// sketch starts

void setup()    {
// Set up our I/O pins and our serial port to
// talk to the Arduino IDE.
 Serial.begin(57600);
  Serial.print("SONAR\n");

  pinMode(powerPin, OUTPUT);
  pinMode(echoPin,OUTPUT);
  digitalWrite(powerPin,HIGH);
  digitalWrite(echoPin,LOW);
}

// the loop() method runs over and over again,
// as long as the Arduino has power

void loop()
{
  long time = 0;

  digitalWrite(powerPin, LOW);
       // turn the SONAR board on
  delay(200);
       // wait for board to stabilize
  pinMode(echoPin,INPUT);
       // pull init pin high
  time = pulseIn(echoPin,HIGH);
       // Get the echo duration time
  time-=200;
       // subtract out some delays

  // I fudged some here, your adjustment
  // may vary...
  Serial.print(time/149);
  Serial.print(" in ");
  Serial.print(time/59);
  Serial.println(" cm");

  digitalWrite(powerPin,HIGH);
       // Turn power off
  pinMode(echoPin,OUTPUT);
       // Make the ping/echo pin an output
  delay(1000);
       // Delay a second for next pulse
}
```

to mate with the connector. That is why the SONAR board connections are called out as U2 pin numbers. See **Figure 2** to locate U2 on the board.

Okay, that takes care of the hardware ... now for the code. The Arduino does a very good job at hiding the complexity of embedded programming. This means that what would normally be a complicated program using interrupts, hardware timers, Data Direction Registers, and the like turns out to be very simple.

Listing 1 is the entire SONAR program. This program gives the range in both cm and inches; you can choose your favorite unit of measure to use. I removed 200 µs from the time returned because I noticed that my ranges were about two inches (5 cm) too long (using my tape measure held in two hands while aiming). This problem occurs because I use C1 to delay the rise of the *Init/Echo* line which causes the pulse to stretch a little bit there. Also, the conversion that I use to calculate the range is based on the speed of sound at sea level at 15 degrees C. I live at 5,000 feet and my room was closer to 25 degrees C. Combining my inaccurate measurement technique, air pressure, temperature differences, and my delay capacitor together — along with some amount of code latency — means that my times are a little off. You'll need to experiment to see what error the system introduces when you build this setup. I have typically found that the Polaroid boards hacked from cameras are a little off — some of this error may even be in the board itself. Regardless, after you apply a little *fudge factor* you'll find that the SONAR boards are a handy addition to many projects.

I don't think these cameras have been made in nearly 20 years, but you can still find them at thrift stores for a small cost and even eBay has them all frequently for between $4 and $10.

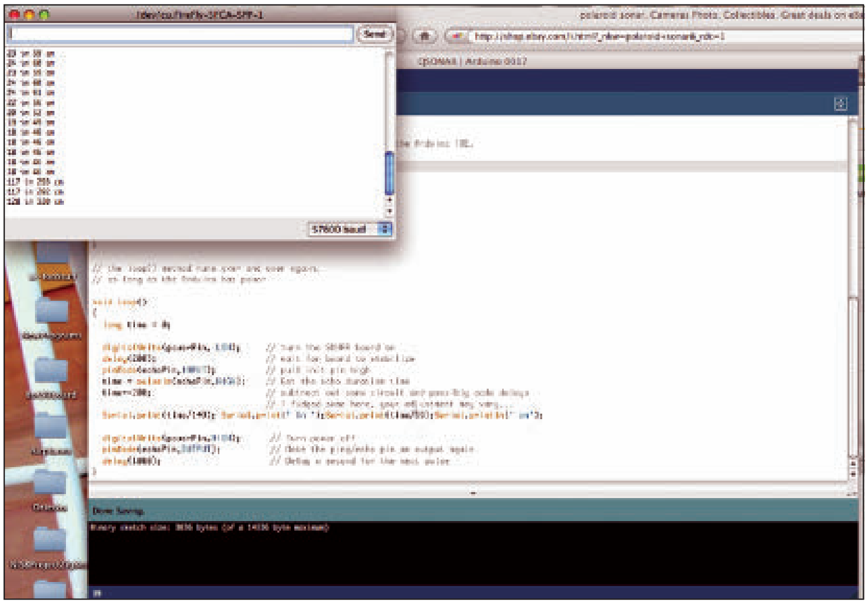


Figure 3. Using the serial monitor in Arduino.

The code in **Listing 1** does a SONAR cycle every second to send the code to the serial port that you can configure and run in the Arduino IDE (see **Figure 3**). This is a convenient way to know what your program is doing. My layout uses a SparkFun Bluesmirf Gold Bluetooth connection, but just a serial cable for the typical Arduino board works fine.

I hope you've learned something this month or at least been entertained in the process. I enjoyed getting back to this old project and I hope that I may have re-invigorated you to look into your closet and pull out that dusty old Polaroid and look at it in a different light. Next year, I'm planning to enter the SparkFun AVC with a ground based vehicle. I'm not brave enough to risk a flying machine just yet! (See the **sidebar** on this year's event.) As usual, if you have any robotics questions don't hesitate to let me know at roboto@servo magazine.com. I'll do my best to answer you! **SV**

Autonomous Vehicle Competition

In the Ides of April, SparkFun in Boulder, CO, held their second Autonomous Vehicle Competition. It was well attended and well played. The competitors were trying to get their autonomous robots to circumnavigate the SparkFun building in the fastest time. There were two classes: ground-based and flying. The competition included five aerial and 13 rolling robots. The audience, as well as the competitors, had fun. Everyone was treated very well by the SparkFun staff.

This competition was a blast to watch and listen to (more on that later) because there was such a wide variety of solutions to root for! All of the aerial entries were fixed wing (no helicopters) and made of foam. Since foam is more durable, easy to fix, and fairly inexpensive.

Here are the results of the competition:

Aerial Category:

1. Robota, -4 seconds
2. Death by Pinetree, 3 seconds
3. UofA Robotics, 5 seconds
4. DIY Drones, 37 seconds
5. Donuts, Coffee, Muffins, 30% complete

Ground Category:

1. Team Tobor, 1 minute 55 seconds

Autonomous Vehicle Competition continued ...

Figure A. Elmo was popular! Autobahn finished third.

Figure B. Ground-based winner, Team Tobor.

2. Bon Scott, 3 minutes 12 seconds
3. Autobahn, 3 minutes 14 seconds
4. Autocrusher, 3 minutes 27 seconds
5. BlueBot, 50% complete
6. Buzzbut, 30% complete
7. Project 240, 25% complete
8. Geeknight Avenger, 25% complete
9. Jaunty Python, 25% complete
10. Sharcbot, 22% complete
11. Tinkerbell, 20% complete
12. B.O.B., 10% complete
13. Diminished Expectations, 3% complete

There were also some special awards given to competitors whom the judges felt deserved recognition for efforts or results above and beyond the call of sleeplessness:

Engineer's Choice Award: University of Arizona Robotics
Kill Switch Award: B.O.B.
Rookie Award: BlueBot
Best Dressed: SharcBot

Figure D. Aerial winner, Team Robota.

Figure E. The SparkFun employee band: Miles Per Gallon.

You might have noticed that the winner of the aerial competition had a *negative* time. Some explanation of the rules is in order to explain this oddity. For the flying competitions there were two special rules that could modify your time:

1. If you land autonomously, 15 seconds is deducted from your time.
2. If you land autonomously within the yellow tape rectangle in front of the judges stand (loading dock), 30 seconds is deducted from your time.

The first, second, and third place teams all landed autonomously. Team Robota landed in the taped rectangle and it was pretty amazing to watch!

When the judges were busy, SparkFun kept the crowd entertained with their company's impromptu band (**Figure E**). These guys sounded great and kept the energy levels up when folks might otherwise have gotten bored and wandered off in between events.

There were a few humorous moments in the competition. The robot B.O.B. (Bounces Off Bumpers) had an electrical failure resulting in some smoke which required a quick application of a "kill switch" (earning him the Kill Switch Award), which prompted the band to declare a possible name change to B.I.F. (Bursts Into Flames). One of the aerial entrants searched for his flight path, but apparently got bored and flew off towards the foothills to the west of us. We learned later that the robot landed on the roof of the museum down the street. The plane was damaged during the landing, but the team came prepared with a spare. Another "interesting" event occurred when one of the robots landed on the roof of a nearby military aerospace contractor. This was a more touchy moment and required calls to Washington, DC to get permission to remove the plane from the roof of the facility. The contestants did get their plane back and went on to have more tries at the prize.

Figure C. Everybody go!

NEW PRODUCTS

SERVOS/CONTROLLERS

Monster Torque Servos

Hitec now offers two of their most technologically advanced "Monster Torque" servos.

Both of these powerhouse servos have premium performance and durability. Each features a 7.4V optimized coreless motor, integrated heatsink case, and a top case with two hardened steel gear pins supported by brass axial bushings. These servos are designed with Hitec's newest high resolution "G2.5" 12-bit programmable digital circuit and indestructible titanium gears. The HS-M7990TH utilizes a high resolution magnetic encoder while the HS-7980TH employs a conventional potentiometer.

The HS-7980TH and HS-M7990TH are ideal for high performance giant scale aircraft, truggies and buggies, monster trucks, and high performance boats.

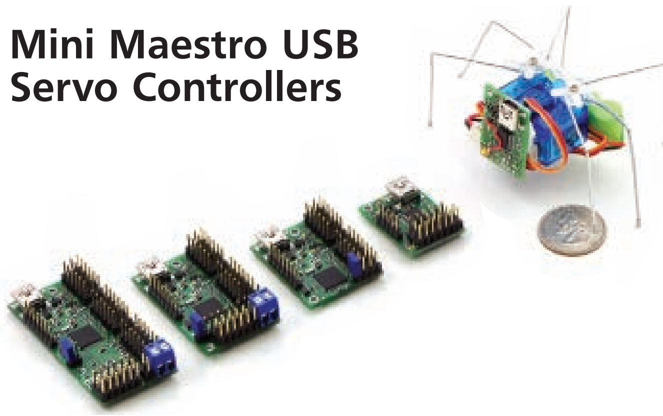
Common Features:

- G2.5 12-bit Digital Programmable Circuit
- Titanium Gear Train (MK first gear)
- Ultra Performance Coreless Motor
- Integrated Heatsink Case
- (8) O-Rings for Water/Dust/Fuel Protection
- Dual Ball Bearing Supported Output Shaft
- Standard Plus Sizing (10% larger than a HS-7955TG)

For further information, please contact:

Hitec	Website: **www.hitecrcd.com**

Mini Maestro USB Servo Controllers



Pololu has released the Mini Maestro 12-, 18-, and 24-channel USB servo controllers. In addition to a TTL serial interface, these small boards incorporate native USB control for easy connection to a PC and programmability via a simple scripting language for self-contained, host controller-free applications. The Maestros' extremely precise, 0.25 us resolution servo pulses have a jitter of less than 200 ns, making these servo controllers well suited for high-performance applications, and individual speed and acceleration control for each channel allow for smooth, seamless movements. The Mini Maestros feature configurable pulse rates up to 333 Hz and can generate a wide range of pulse widths to allow maximum responsiveness and range from modern servos. Units can be daisy-chained with additional Pololu servo and motor controllers on a single serial line.

A free configuration and control program is available for Windows and Linux, making it simple to configure and test the board over USB, create sequences of servo movements for animatronics or walking robots, and write, step through, and run scripts stored in the servo controller. The 8 KB of internal script memory allows storage of up to approximately 3,000 servo positions that can be automatically played back without any computer or external microcontroller connected.

The Maestros' channels can also be used as general-

purpose digital outputs and analog or digital inputs, providing an easy way to read sensors and control peripherals directly from a PC over USB. These inputs can be used with the scripting system to enable creation of self-contained animatronic displays that respond to external stimuli.

The unit prices are $29.95, $39.95, and $49.95 for the Mini Maestro 12 (item #1352), 18 (item #1354), and 24 (item #1356), respectively. Kit versions are available for applications with tight size or weight restrictions.

For further information, please contact:

Pololu Corporation

3095 E. Patrick Ln. #12
Las Vegas, NV 89120
Tel: **877•7•POLOLU** *or* **702•262•6648**
Fax: **702•262•6894**
Email: **www@pololu.com**
Website: **www.pololu.com/maestro**

BREADBOARDS

Economy Prototyping Breadboards

Global Specialties, a leader in electronic training systems and prototyping and design products, introduces a new family of products to its economy line of solderless breadboarding systems. The product grouping — which includes four RoHS compliant products — utilizes Global's new highly durable ABS transparent breadboarding sockets. The transparent ABS sockets allow for easy viewing of prototyping connections which enables the reduction of circuit construction time and verification making them perfect for educational applications and beginners in electronics.

The GS-100T transparent bus sockets (also included on the GS-830T and PB-101T) have red and blue continuity indicators to allow for reliable signal distribution. The new products include the GS-100T (100 tie points), GS-630T (630 tie points), and GS-830T (830 tie points) solderless breadboards, and the PB-101E Proto-Board™ with 1,360 tie points and aluminum backplate.

For further information, please contact:

Global Specialties

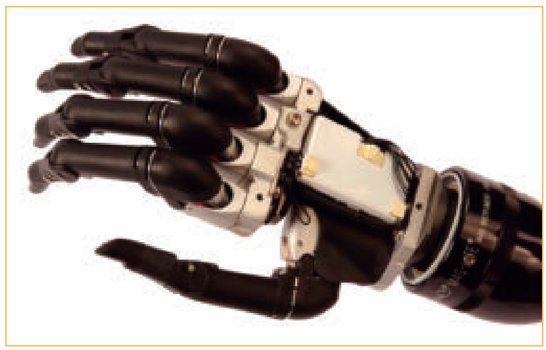
Website: **www.globalspecialties.com**

PROSTHESES

New i-LIMB Pulse Unveiled

Touch Bionics, developer of advanced upper-limb bionic technologies, has launched the i-LIMB Pulse: an all-new version of the revolutionary i-LIMB Hand — the world's first commercially available bionic hand. The i-LIMB Pulse is a significant advance for the i-LIMB product line, with a host of enhancements including pulsing grip strength, software-enabled grip patterns, and robust aluminium features for improved strength.

The i-LIMB Pulse joins a family of products that has been fitted to more than 1,200 patients worldwide. The i-LIMB Pulse is offered in addition to the existing i-LIMB Hand, and both products are available to customers depending on the patients' preferences. Its new features have been driven by Touch Bionics' experiences in the marketplace as a pioneer of bionic technology and clinical support.



"Having experienced over three years in the market with the i-LIMB Hand, we have gathered unparalleled insight into the needs and requirements of users of upper extremity prosthetic devices," said Stuart Mead, Touch Bionics' CEO. "This has contributed significantly to the development of the i-LIMB Pulse — for example, we know that the majority of users, whether male or female, prefer a device with natural body lines, so we invested considerable time in miniaturizing components and internal structures to offer two sizes that will accommodate almost all user preferences."

The i-LIMB Pulse is so called because it utilizes Touch

Bionics' patented pulsing technology to provide increasing and controllable grip strength. When the i-LIMB Pulse closes on an object, the user has the option to use the pulsing feature to apply significant additional grip force, allowing the device to grasp an object more tightly. These high-frequency electronic pulses are very important in everyday activities for users, such as tying shoelaces or doing-up a belt.

With an aluminium chassis, the i-LIMB Pulse's robust design features make it Touch Bionics' toughest prosthetic device yet, capable of carrying a load of up to 90 kg.

"The i-LIMB Pulse is a very exciting technology development, particularly for someone like myself, who is looking to get back to a level of duty in the fire service," said Ian Reid, a firefighter from Thurso in Scotland, who lost his right hand in a tragic holiday bus crash in 2003, and the first person to be fitted with an i-LIMB Pulse. "The pulsing effect, increased robustness, and range of grip features will hopefully give me the increased level of function I'm looking for."

With a range of automated features that allow various combinations of grip patterns and other digit-postures to be activated by the user, the i-LIMB Pulse offers an unparalleled degree of flexibility to its users. Patterns like index point, precision pinch, lateral key grip, and three jaw chuck (tripod) are all now easily activated in a single action by the user.

Key to the i-LIMB Pulse is BioSim, Touch Bionics' new Bluetooth-enabled software that allows prosthetists and — for the first time ever — users (using MyBioSim) to select the features and control strategies that work best for them.

For prosthetists, BioSim is a complete software toolset that allows clinical practitioners to customize the i-LIMB Pulse to the specific needs of the user. It allows real-time assessment of users' myoelectric impulses with the ability to adjust gain and threshold settings, select different control strategies, and enable or disable features and grips, including the pulsing effect.

BioSim also provides statistical analysis and documentation of the device wearer's usage patterns — a feature designed to improve patient outcomes and training.

The i-LIMB Pulse began shipping June 1, 2010, and Touch Bionics is currently accepting orders on the product.

For further information, please contact:

Touch Bionics Website: **www.touchbionics.com**

SOFTWARE

RobotBASIC 4.1 Now Available

RobotBASIC has released Version 4.1 of their free robot control language. New in this version are parameterized subroutines with local variables making it easy to create includable libraries and pseudo object-oriented code. Multimedia extensions have been added to allow control of a new audio/video window and to record audio files under program supervision.

C-style syntax (i.e., ++, !=, &&, +=, etc.) can now be used to enhance the programming experience and to allow students a smooth transition from the ease of BASIC to the more cryptic nature of C.

Many enhancements have been added to the HELP system, the graphics engine, and the support for USBmicro I/O boards. The user has been given more control over the output window and variable manipulation.

Even with all the improvements, RobotBASIC still requires no formal installation and can be run from a CD or USB drive as either an interpreter or compiler.

For further information, please contact:

RobotBASIC Website: **www.RobotBASIC.com**

KITS

Low Capacitance DMM Adaptor

Many modern multimeters come with capacitance ranges, but they're not helpful for very small values. Jaycar Electronics has a new kit that is an adaptor that allows a standard digital multimeter to measure very low values of capacitance from less than one picofarad to over 10 nF. It allows measurement of tiny capacitors or stray capacitances in switches, connectors, and wiring. The kit is complete with PCB, components, and case. Simply add a 9V battery and just about any modern DMM.

For further information, please contact:

Jaycar Electronics Website: **www.jaycar.com.au**

EVENTS Calendar

ROBOTS.NET

Send updates, new listings, corrections, complaints, and suggestions to: steve@ncc.com or FAX 972-404-0269

Know of any robot competitions I've missed? Is your local school or robot group planning a contest? Send an email to steve@ncc.com and tell me about it. Be sure to include the date and location of your contest. If you have a website with contest info, send along the URL as well, so we can tell everyone else about it.

For last-minute updates and changes, you can always find the most recent version of the Robot Competition FAQ at Robots.net: **http://robots.net/rcfaq.html**

— R. Steven Rainwater

JULY

7-11 BOTBALL National Tournament
Edwardsville, IL
Autonomous robots built from standardized kits must move black and white balls around a game board.
www.botball.org

11-15 AAAI Mobile Robot Competition
Atlanta, GA
This year's event is the Semantic Robot Vision Challenge — a scavenger hunt for robots. Autonomous robots are given a list of objects. First, they must learn what the objects look like by searching the Internet for photos, then they must explore the local environment to find the objects.
www.aaai.org/Conferences/conferences.php

13-18 AUVS International Underwater Robotics Competition
Space and Naval Warefare System Center
San Diego, CA
Autonomous underwater vehicles must complete an underwater course with various requirements. Bots cannot be greater than six feet long by three feet wide by two feet deep, and not be greater than 100 kg mass.
www.auvsi.org/competitions/water.cfm

17-20 RoboBombeiro
San Miguel Pavilion, Guarda, Portugal
Autonomous fire fighting robot contest.
http://robobombeiro.ipg.pt

18-23 CIG Car Racing Competition
Barcelona, Spain
Race cars controlled by neural networks created by evolutionary algorithms.
www.wcci2010.org

19-23 K'NEX K*bot World Championships
Las Vegas, NV
Events for two wheel autonomous robots, four wheel autonomous robots, and three motor remote control K*bots.
www.kbotworld.com

24 Chibots RoboMagellan
Chicago, IL
Autonomous robots must do waypoint navigation and visual object recognition.
www.chibots.org

AUGUST

6-8 Rescue Robot Contest
Kobe, Japan
Autonomous and teleoperated search and rescue robots.
http://rescue-robot-contest.org

9-13 AUVS International Aerial Robotics Competition
Puerto Rico
Autonomous aerial robots and sub-vehicles compete in a series of tasks.
http://iarc.angel-strike.com

SEPTEMBER

3-6 DragonCon Robot Battles
Atlanta, GA
Autonomous and remote control robots battle at the popular science fiction convention.
www.dragoncon.org

5-12 Microtransat Challenge
County Kerry, Ireland
Autonomous sail boats start on a race across the

Atlantic ocean; this will take three to four months to complete.
www.microtransat.org

18 **Robotour**
Bratislava, Slovakia
Autonomous robots must perform a navigation task in a park.
http://robotika.cz

OCTOBER

3-10 **Devyanin Mobile Robots Festival**
Moscow State University, Moscow, Russian Federation
Autonomous robot race.
www.mobilerobots.msu.ru/en

22-24 **Critter Crunch**
Hyatt Regency Tech Center, Denver, CO
Robot combat — 2 lb and 20 lb event catagories. Autonomous and Remote Control.
www.milehicon.org/critrule.htm

NOVEMBER

2-3 **Junior Robotics Challenge**
Singapore
Line following can collection.
http://jrc2009.webs.com

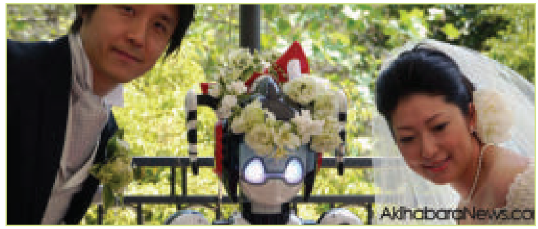
7 **International Micro Robot Maze Contest**
Nagoya University, Japan
Micro Robot Racer (1 cm cube), Climbing Competition (1 cm cube), Maze Solver (1 inch cube), Two Leg Robot Competition (2 inch).
http://imd.eng.kagawa-u.ac.jp/maze

bots IN BRIEF

HERE COMES THE BOT

Robot-themed weddings are becoming more and more popular (at least in Asia), from robots modeling wedding gowns to serving as ushers. Now we have the world's first robot marriage witness. It was a natural fit for the bride, who met the groom while he was working on robots at the Nara Institute of Science & Technology. So, Kokoro's I-FAIRY communication robot played a prominent role in declaring the wedding a success and wishing the couple a happy future together. Kokoro does not rent the I-FAIRY so must have allowed this special appointment since the free advertising helped widely promote the company (which is already famous for its Actroid humanoid robots).

An I-FAIRY costs $70,000 USD, and two of five units produced have already sold since its unveiling at CES 2010 in January. Perhaps one of Japan's "wedding centers" will buy one and make robot-themed weddings available to the masses.



DRONE-ING ON

With all the fuss lately concerning drones in the military, isn't it nice to know that the next generation is on its way? Boeing recently unveiled its Phantom Ray — part of their Phantom Works division.

SPACING OUT

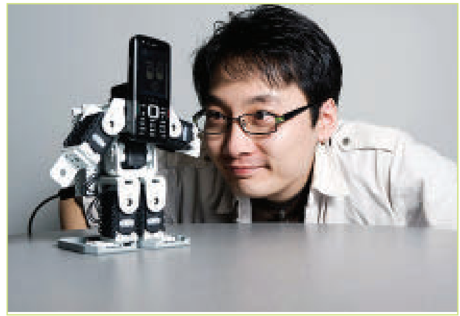
SOHLA (Space Oriented Higashiosaka Leading Association) is planning to send a humanoid robot to the moon by 2015. At a cost of about $10.5 million, the Japan-based group believes this can help stimulate the local economy by getting smaller organizations involved.

LET JULIA ENTERTAIN YOU

The Advanced Control Lab at National Taiwan University is working on JULIA — an interactive humanoid that works with touch screen or voice commands, and has a display for home media information. She can be used as a security guard when you are at work and, when you get home, entertain you by singing and dancing.

bots IN BRIEF

MEET THE CALLOS



Ji-Dong Yim and Chris Shaw — scientists at Simon Fraser University's School of Interactive Arts and Technology (SIAT) — are the proud parents of a robotic cell phone family that can walk, dance, and express human-like emotions.

Yim, a doctoral student, and Shaw, an associate professor, first used cell phone technology to create Cally: a physically active robotic cell phone that stands roughly 16 centimeters high. She walks, dances, and mimics human gestures. She can also help cell phone users make electronic eye contact with the person to whom they are talking to by tracking human faces.

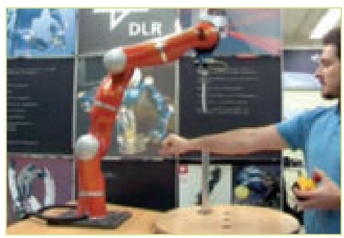
The SIAT researchers have most recently used wireless networking, text messaging, and other interactive technologies to give birth to Callo. He is taller and more emotionally sophisticated than his older sister. Callo's viewing screen registers text-messaged emoticons as human-like facial expressions. His robotic shoulders can slump and his arms can start waving frantically if he's interactively triggered to respond to an emotional crisis (such as relationship break-up).

"Imagine you are video-calling with me through Callo," explains Yim. "When you move your robot, my robot will move the same, and vice versa, so that we can share emotional feelings using 'physically smart' robot phones."

BRAVE BOT

One of the unsung heroes ofthe recent Times Square incident was the NY Bomb Squad's robot that was sent in to check out the situation, break into the SUV via a window, and help disarm the crude device.

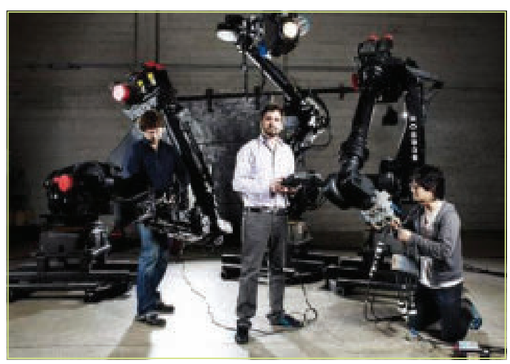
ARMED AND LETHAL?



At the recent 2010 IEEE International Conference on Robotics and Automation, the German Institute of Robotics and Mechatronics programmed a robotic arm to stab a silicone lump, a dead pig's leg, and some human volunteers. It was armed with steak and kitchen knives, a screwdriver, and scissors. When the safety was off, the bot inflicted cuts on the lump and leg that were deemed potentially lethal. Fortunately, the humans only were used when the safety system was turned on, reminding us you never know when your robot will turn on you.

LIGHTS, CAMERA, ROBOTS!

Robots may be getting laid off too left and right due to the lousy economy, but at least they're easily retrainable. As long as you've got some programming know-how, you can get them to do all kinds of exciting things. One company in particular called Autofuss has taken a trio of ex-industrial six axis Fanuc s430iL robots and turned them into professional camerabots. As one would expect, they're very fast and extremely precise, and they can produce spectacularly smooth and complex shots over and over without screwing up or complaining.



Recently, the Autofuss robots (they're named Puck, Gilda, and Rosie) worked on an ad for Louis Vuitton. The robots aren't doing anything fancy, but the commercial has Sally Ride, Buzz Aldrin, and Jim Lovell in it, so that's pretty awesome all by itself.

Cool tidbits herein provided by Evan Ackerman at www.botjunkie.com, www.robotsnob.com, www.plasticpals.com, and other places.

IN A NANNY STATE

China's Siasun Robot & Automation is developing a 2'7", 55 lb bot that can chat it up on an eight hour shift after a two hour charge. She can also check email, text, check for gas leaks, and call the police in case of an emergency. The new family Nanny is due out in 2015 at a price of approx. $1,465.

HUB-A-HUBBA

These brightly colored robots double as two port hubs when you pop open their front plates. Once you have finished using them practically, you can wind them up for a few minutes and their arms and legs move. At a size of 7.5 x 9.7 x 4.3 cm, these minibots are compatible with MAC OS X (8.1 or later) and Windows 98/SE/2000/XP and cost about $27.

GOING DEEP

Apparently, South Korea's Ministry of Land, Transport, and Maritime Affairs is planning to invest $20 billion won in an underwater robot project. The project involves about the construction of an "articulated, multiple-mobile" robot which will be capable of handling various tasks at a maximum of 6,000 meters underwater, ranging from exploration and rescue operations to such environmental missions as maintenance of underwater flora. The development of the robot with capabilities to operate in shallow water is the first phase of the project and scheduled to be completed by 2012. During the second phase from 2013 to 2015, the robot will further be improved to perform its planned deep water tasks.

The Korean Ministry is also developing a wireless aquatic robot that can swim at a speed of 18 meters per second and crawl at a speed of 30 mi/sec. Each one has six paddles and a camera to find sunken ships. The $17.88 million project hopes to have a prototype by 2012 that can reach a depth of up to 200 meters, and another by 2015 that can reach a 6 km depth.

IT'S ALL IN YOUR MIND

Using a combination of a handful of hardware, a bundle of software, and a bucket of brains and ingenuity, one intrepid engineer decided to control his Rovio … with his MIND! Robert Oschler has taken the Rovio to the next level, and is able to control it remotely, hands-free, with nothing but his facial movements and concentration. (And a fancy $300 Emotiv EPOC Neuroheadset.)

Here's how it works: Oschler wears his Emotiv EPOC headset which has electrodes to detect his brain waves, as well as two gyroscopes to detect movement. The headset communicates with Roboclient which interprets the signals, and sends them wirelessly over Skype to another computer. The other computer reads the Skype data and sends it to the Robodance software. Robodance turns the signals into robot commands which are sent via WiFi to the Rovio.

Pretty darn cool.

FIDDLING AROUND

Toyota's reclusive artistic genius — the Violin-Playing Robot (not seen since its unveiling in 2007) — is making its second major public appearance at the Shanghai World Expo 2010 as part of the Japanese Pavilion. Rather than playing the same tune as before, Toyota has reprogrammed the robot to skillfully play a song that is distinctly Chinese.

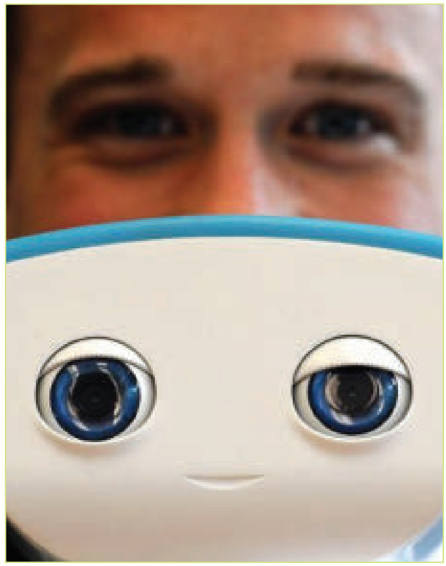
BEAR WITH US

Fujitsu's new fluffy bear invention has a camera in its nose that can detect human faces and actions (such as waving of hands) while sensors inside its head and limbs can detect human touches and caresses.

The bear can respond with more than 300 actions of its own, from giggling and laughing to waving its paws and taking a nap — even snoring.

"We want to offer an object that can become part of the family, nursing home, or school, and that can benefit humans," a Fujitsu researcher explained. "We really want it to look natural."

The bear has 12 degrees of freedom, allowing it to move both arms and legs, as well as tilt its head and move its eye brows and ears. A total of eight touch sensors are located throughout its body allowing it to react to being petted, and two sensors in its arms detect when someone is shaking its hands. Gyroscopes and accelerometers detect when the bear is being moved around. Fujitsu has been testing it at several medical institutions, but it seems commercialization is going to be a little ways off as the face recognition isn't working as reliably as they want.



NEW FOOD POLICE

Meet Cory Kidd, CEO of Intuitive Automata, at his office in Hong Kong next to a robot he designed as a dietary assistant. Users can have daily conversations with the 38 centimeter tall (15 inch) robot, which will crunch calories and provide feedback and encouragement on their weight-loss progress.

GET SCOOPED

Kikuchi Manufacturing is showing off their "Mini Robocue," a smaller version of a similar human-scooping rescue robot called "Robocue" that was first unveiled in 2009. This second generation is considered a "mini" because of an overall reduction in size and weight compared to the first generation.

Mini Robocue: 2,310 mm x 810 mm x 1,450 mm (L,W, H); 350 kg

Robocue: 1,900 mm x 1200 mm x 1,600 mm (L,W, H); 1,500 kg

While a bit longer, it is significantly lighter and more compact in three dimensions. The robot also has improved mobility thanks to its independent front and back tank treads. This increases its turning circle, allowing it to swivel in place. The idea was to allow the robot to climb steps inside of a building and still be able to turn when it reaches a landing. People in need of rescue are loaded onto the conveyor belt which scoops them up into the robot's body. (There is a size limit, however.)

The robot can be controlled remotely over a monitor via a 100 m cable, or controlled from a 50 m via radio signal if the operator has a clear view. It can also be equipped with manipulators for dealing with potentially hazardous materials. Kikuchi Manufacturing is hoping to sell at least one unit to each of the fire stations across Japan, so they're aiming at a price point of around $16,000 USD.

COMBAT ZONE

RIOBOTZ COMBOT TUTORIAL SUMMARIZED

HammerBots

● Summarized by Kevin M. Berry

Professor Marco Antonio Meggiolaro, of the Pontifical Catholic University of Rio de Janeiro, Brazil, has translated his popular book — the *RioBotz Combot Tutorial* — into English. In May's Combat Zone, *SERVO* summarized the "LaunchBot" portion of his exhaustive chapter on weapons. Chapter 6 — Weapon Design — is a college level textbook on the design and operational theory of today's combat weapon systems. In this article, we present a much simplified version of the HammerBots section of this chapter. Marco's book is available free for download at **www.riobotz.com.br/en/tutorial.html**, and for hard copy purchase (at no profit to Marco) on Amazon, published by CreateSpace. All information here is provided courtesy of Professor Meggiolaro and RioBotz.

HammerBot Design

Hammers usually need to be pneumatically powered to be effective. This is because they have to reach their maximum speed in only 180 degrees of rotation. Since most pneumatic actuators are linear cylinders, you'll need some type of transmission to convert linear into rotary motion. This can be done

in several ways. One of the lightest solutions — adopted by the super heavyweight The Judge — is implemented using a pair of opposing heavy-duty chains (colored in red and blue in **Figure 1**). When the right port of the cylinder in the figure is pressurized, it makes the piston move to the left and pull the red chain which generates a rotary motion in the hammer.

The hammer can have a spring mechanism to move back to its starting position after an attack. The best solution, though, is to have a double-acting cylinder to retract the hammer at high speeds, with the aid of the blue chain shown in the **photo**. This allows the hammer to get ready in less time for the next attack. Also — and most importantly — it guarantees enough torque to the hammer in both directions to work as a self-righting mechanism in case the robot gets flipped upside down.

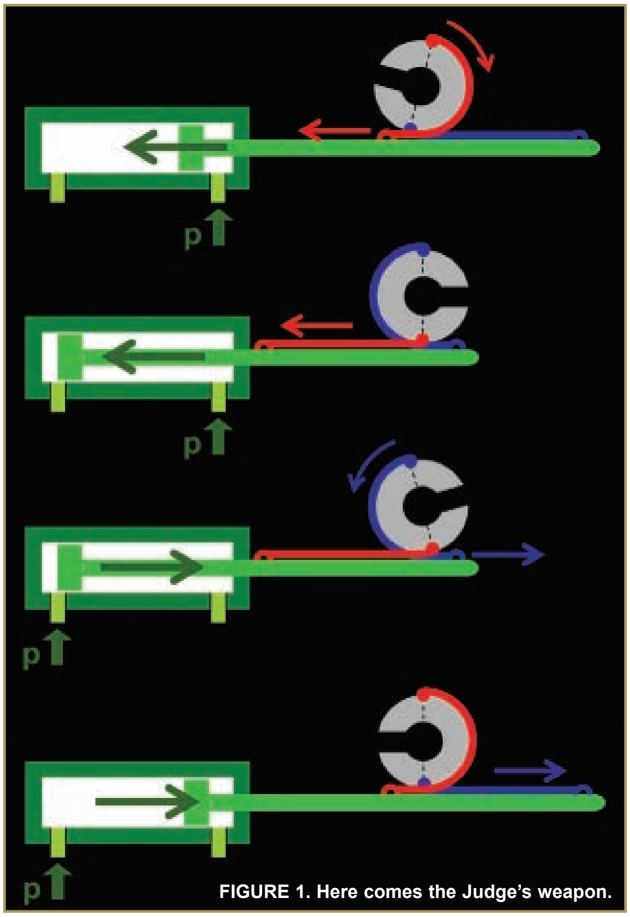


FIGURE 1. Here comes the Judge's weapon.

Hammer Energy

No matter which mechanism you use to generate a rotary motion, it is not difficult to estimate the energy and the top angular speed of the hammer in a pneumatic robot. If we assume no energy loss due to friction or pneumatic leaks, then the energy delivered by the cylinder is approximately equal to its operating pressure times its internal volume. If the hammer has much more inertia than the cylinder piston and the transmission mechanisms, then we can say that this energy is entirely converted into kinetic energy in the hammer.

Through a rather long — but not particularly difficult — mathematical proof, Prof Mathmatico, er, make that Meggiolaro, shows that it is slightly better to design the transmission system such that the hammer hits when the piston is extended or "pushing," rather than retracting or "pulling." Depending on the transmission design, this might place the cylinder in the front of the robot (more exposed to attacks) and limit the reach of the hammer head. These are details best understood by reading section 6.7.1 of the tutorial.

Hammer Impact

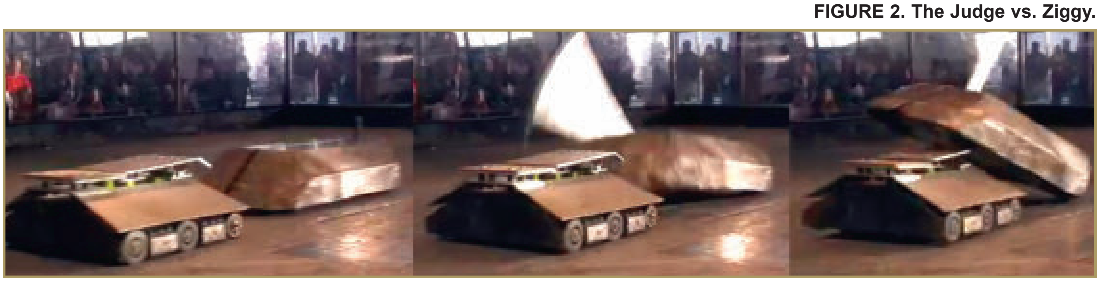
The sample system used for calculations is:

- 1,000 PSI system
- 4" bore cylinder with a 1.25" piston
- 8" piston stroke with 6.5" used before contacting the opponent
- Hammer handle is 36" long with a mass of 15 lb, with a 10 lb hammer head

If we place the cylinder in the back of the robot, using the mechanism from **Figure 1** then the energy from the pulling motion would accelerate the hammer up to 521 RPM, resulting in a hammer head speed of 111 MPH! (Editor's note: This is why The Judge is one of the most feared bots in combat robot history — imagine armoring the TOP of your bot for this magnitude of impact!)

Note that the robot will tend to move backwards during the acceleration of the hammer. Therefore, it needs to compensate for that by braking its wheels. The chassis will also tend to tilt backwards from the reaction force of the hammer accelerating forward. Powerful hammerbots may even see their front wheels lift off the ground because of that, as shown in **Figure 2**. You can see The Judge tilting backwards right before it even touches the opponent. Excessive tilting may leave it vulnerable to wedges or launchers

FIGURE 2. The Judge vs. Ziggy.



that might sneak in underneath (as shown in the right side of **Figure 2**, just before The Judge was launched by Ziggy). To avoid that, it is a good idea to move the center of mass of the hammerbot forward.

The right portion of **Figure 2** shows that the tilting angle of the chassis is increased even more after the hit, due to the reaction impulse from the impact. The speeds after the impact and all the involved impact energies can be calculated from the very same equations used for spinners. Since the attacked robot is hammered against the arena floor, it usually does not move its center of mass, it only deforms due to the attack. This impact problem is similar to an offset bar spinner hitting a flexible but very heavy wall as shown in **Figure 3**.

The attacked robot would then have an infinite effective mass (not exactly infinite, but that of the Planet Earth!), while the hammerbot's effective mass would work like an offset spinner. The speed of the attacked robot after the hammering is, of course, zero. The hammerbot chassis gains vertical speed and may, in fact, spin backwards.

Note that if the back wheels of the hammerbot are still in contact with the ground immediately after the impact against the opponent, then a second impact will probably occur. With the back wheels gaining a downward speed after the initial impulse, they will press against the arena floor and receive a vertical reaction impulse. This back wheel impulse is good for the hammerbot, because it prevents its chassis from tilting backwards too much. The final linear and angular speeds of the hammerbot chassis can be calculated using the same equations from the second impact that happens when a robot is hit by a drumbot or vertical spinner. (These proofs are fully developed in the Tutorial.)

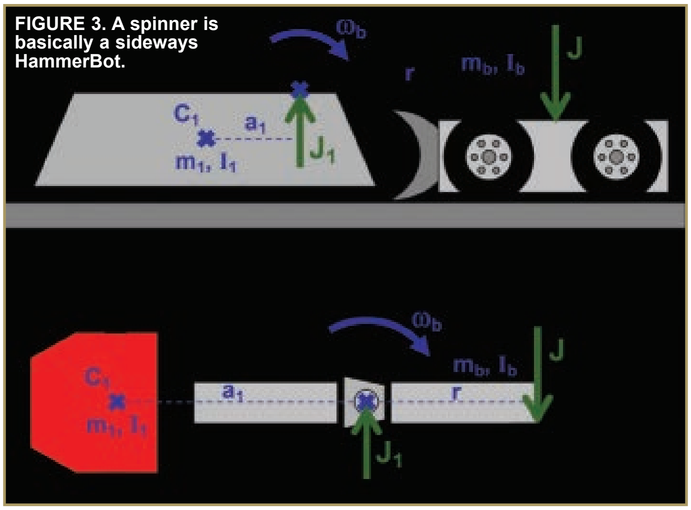


FIGURE 3. A spinner is basically a sideways HammerBot.

Summary

In this mere 1,000 words, I've managed to grossly simplify the treatment of this subject from the Tutorial. A longer article on ThwackBots — a close cousin to HammerBots — is in the offing, which should serve to provide a more thorough understanding of the mechanics of this interesting and scary weapons system. **SV**

BUILD REPORT: A Reintroduction to Wedges

● by Thomas Kenney

In mid 2007, I began the construction of Gilbert — MH Robotics' first antweight. The bot has gone through many revisions over the years, beginning as an inertia-labs kit with sheet metal bent over the top to form a wedge, and going on to become one of the flagships of our fleet of overpowered bricks sporting hinged wedges. The second revision of our hinged wedge design involved a titanium shaft supporting two to three blocks of UHMW or aluminum, to which was secured the wedge piece itself. This general design suffered from a few issues before it was applied to more of our bots; most of these

FIGURE 1. Our original hinged wedge assembly, including the shaft, 7075 aluminum plate, and UHMW blocks.

problems involved the wedge jamming in the downwards position and high-centering the robot. Following this experimentation, the design was applied to our successful beetle Cloud of Suspicion's first two revisions, as well as the third version of Gilbert. It proved effective time and time again against weaponless bots and spinners alike, and I actually wrote an article for *SERVO* about the design over a year ago. Unfortunately, as we began to travel and compete with more effective bots such as the Fierce robot fleet and many east coast spinners, wedgectomies became a steadily increasing occurrence. After having Gilbert's wedge torn off and the top carbon fiber subsequently destroyed along with an even more caustic but successful fight with Cloud of Suspicion against the under-cutter Itsa?, I finally decided that it was time for something new.

FIGURE 2. The third revision of Cloud of Suspicion, sporting our most recent wedge design.

FIGURE 3. Featherweight Pinball, showing its shock mounted steel flat plate.

The first thought that came to mind was to replicate the similar hinged wedge system used on our featherweight, Pinball. Pinball's wedge differed from most of ours at the time in that the steel wedge itself is welded directly to the shaft which sits loose in an oversized hole made in each of the 1" thick UHMW side pieces. An initial problem we encountered with Pinball's wedge design was that the shaft could be torn through the UHMW if pulled hard enough by a vertical spinner. This was later fixed by welding together a .125" 4130 steel bracket that completely covered the extrusion, leaving a hole for the shaft to ride in.

I likely would have gone this route if not for a fight with a horizontal spinner that left Cloud of Suspicion's aluminum wedge support extrusion lopped off. With all this in mind, I chose to follow Mike Daniel's advice and draw up an all new hinged wedge design mildly based off of his old 30 lber, Doom.

The initial problem I wanted to address was how most hinged wedges rely on one or several mounting points based on a flat axis. I chose to combat this (no pun intended) by designing a right angle clamp that holds the shaft down on two perpendicular axis. This makes the entire assembly much more solid and the clamps themselves are nearly impossible to tear off, even though they're being held in through machine screws to the soft UHMW. Of course, to maintain the idea of riding the design of singular plane mounting areas, I chose to have the wedge welded to the shaft. Initially, I planned on an .09" titanium wedge for Cloud of Suspicion and .075" for Gilbert, but the deal fell through.

Fortunately, we were able to go through with the design by using 4130 chromoly steel for the wedges, shafts, and even the clamps which would eventually be hardened for an even stronger assembly. The live shaft sat loose in a slot cut into the top corner of the UHMW, and was kept from falling loose through the several clamps. The only downside to this durable and complex assembly is in how many pieces and separate cuts, bends, and welds it entails. However, in the many fights that it's been through thus far on Gilbert and Cloud of Suspicion, it has proven much more durable by far than the previous design.

Since the creation of Pinball, we realized the effectiveness of a flat, shock mounted steel bumper when used against horizontal spinners. We eventually transferred the idea to the second revision of Cloud of Suspicion with a .04" hardened flat plate (eventually increased to .08" on the current version), and the fourth revision of Gilbert with some plating of the same thickness, and even on our fairyweight, Mango Farmer, using a .09" 7075 aluminum plate. All scales of the design use several "shock dampening sandwich mounts" of various sizes as sold by Mcmaster-Carr (catalog page 1368) for the impact isolation purpose.

In addition to these flat plates, we've recently come up with another alternative to the normal hinged wedge. At the previous RoboGames competition (when we were still using the first version of our hinged wedge design), one of our beetle's wedge support brackets

FIGURE 4. MH Robotics' fairyweight, Mango Farmer with its shock mounted 7075 aluminum wedge-plow.

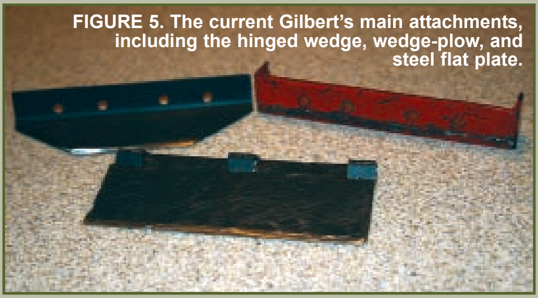


FIGURE 5. The current Gilbert's main attachments, including the hinged wedge, wedge-plow, and steel flat plate.

was lopped off. Before the finals match with a nasty eggbeater, we bent an unhardened steel wedge to about 100 degrees, drilled new mounting holes, and simply screwed it onto the same sandwich mounts used for the steel flat plate. With the tip of the wedge trimmed perfectly against the ground, it held together surprisingly well until the thin .04" steel itself began to bend.

With the slight success of this concept on Cloud of Suspicion, we scaled this up and down for use in all of our wedge bots, and it has actually become the primary wedge for Mango Farmer — which couldn't afford the weight of our hinged wedge assembly. While these have proven successful for the most part, they tend to not work as well against large vertical spinners which usually catch the top corner of the steel, resulting in the wedge 'bot being flipped or pushed back.

To conclude this topic for the second time, I'd like to again state that I don't believe wedges or any bots for that matter should conform to the same standards. The methods that I reference here are only a suggestion derived from a few years of experimentation and combat time. I am more than open to new ideas and always looking for ways to improve the design. SV

COMBAT ZONE'S GREATEST HITS

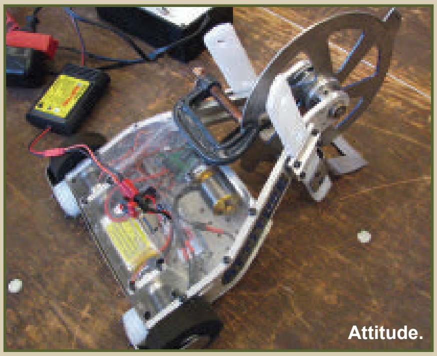
● by Kevin Berry

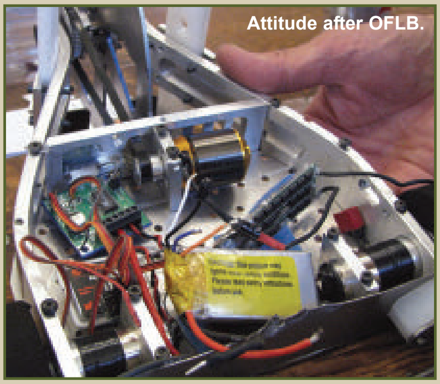
This month, once again, only one builder submitted to Greatest Hits. Come on, bot fighters! There are tons of events going on, and gourmet damage abounds!

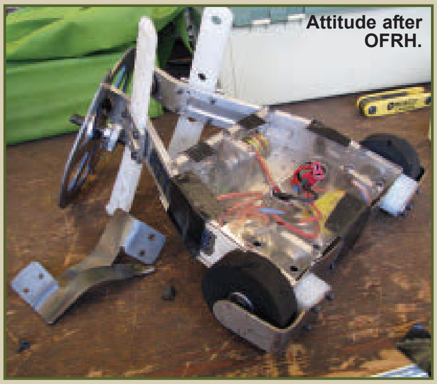
Troy Mock, from Team Asian Invasion, is the Combat Zone Greatest Hits hero for submitting some RoboGames 2010 Greatest Hits. His beetleweight, Attitude, had the dubious honor of fighting two of Gene Burbeck's bots: One Fierce Lawn Boy and One Fierce Round House. Despite the post-battle appearance of Attitude in these pics, Gene said Troy snapped the belt on OFLB, and on OFRH he ripped off some tape and scratched his back armor! In the currency of One Fierce, that's some Great Hits!

Calling All Casualties

Before and after photos, a brief description of the fight, and the builder's name can be submitted to me at LegendaryRobotics@gmail.com. Or, if you have an action shot that clearly shows what's going on, those are welcome too! These don't have to be current, and anything you can (legally) submit — clear back to the good old days of wooden bots and iron builders — is fair game. SV


Attitude.


Attitude after OFLB.


Attitude after OFRH.

EVENTS

Completed Events from Apr 10 – May 9

BotsIQ 2010 Nationals were presented by BotsIQ in Miami, FL, April 16th to 18th.

RoboGames 2010 were presented by ComBots in San Francisco, CA, April 23rd to 25th.

Upcoming Events for Jul – Aug 2010

UBAYA RoboGames 2010 will be presented by the Universitas Surabaya in Surabaya, Indonesia, on July 31 to August 1, 2010. Go to **www.elektroubayarg.com** for more information.

ELEKTRO UBAYA RoboGames 2010

Roaming Robots will hold events July 17th to 18th at RAF Fairford, UK; and July 23rd to 25th at the Farnborough Air Show. Go to **www.roaming robots.co.uk** if you want more information.

Schiele Museum "Clash of the Bots" will be presented by Carolina Combat Robots in Gastonia, NC on July 24,2010. Go to **www.carolinacombat.com** for more information. **SV**

EVENT REPORT: Power Play! Robot Rumble 2010

● by Pete Smith

The Museum of Life and Science, in Durham, NC, hosts this one day event each spring. Last year, three local teams got together to put on a "show and tell" about combat robotics as part of the museum's much bigger robot-themed event.

This year, Carolina Combat (**www.carolinacombat.com**) and Kitbots (**www.kitbots.com**) had hoped to run an insect weight combat event and also stage the first Bot Hockey competition on the East Coast. A ban on lithium polymer batteries anywhere on museum property (other than in cell phones and laptops) turned the Bot Hockey competition into a demonstration-only of the sport with a lot of visitor participation.

Bot Hockey was conceived of by Chris Baron at Robot Power (**www.robotpower.com**) as a way of combining the excitement of ice hockey with the fun of robotics. The rules can be found at **www.bothockey.com** There are two weight classes in the rule set, but only the "Senior" class of 15 lb, 18" square bots has gained popularity. From a start in 2007 with only six "demo" bots designed by myself and built by Chris, the sport is gaining popularity, especially on the West Coast and in Brazil with the result that nine teams entered this year's RoboGames.

Chuck Butler of Carolina Combat has been instrumental in bringing Bot Hockey to the East Coast. He has built a set of four Hockey Bots (see the one on the right in **Figure 1**) and built the 20' x 12' portable arena. I converted my three Sumo bots by adding "wings" to retain the puck and set them up to run in reverse (see the one on the left in **Figure 1**). Chuck's bots were a little overweight and mine were a lot under, but they sufficed for the demo at the museum.

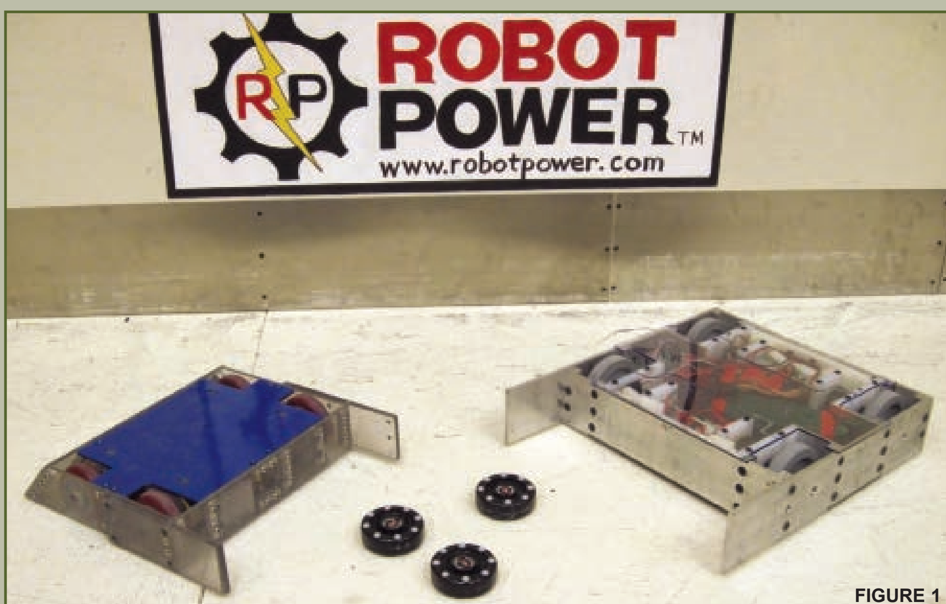


FIGURE 1

FIGURE 2

We set the arena up the evening before the event, and also set up a static display of some of our combat bots and the large heavyweight, "Vera", from Team Moon.

The public was allowed in at 10:00 am on Saturday morning, and the Combat Robots and the Bot Hockey were an immediately popular attraction with crowds lining up three deep to see the matches (**Figure 2**). A Bot Hockey match is usually 10 minutes long but since this was just a show, we ran two minute matches, once every five minutes for thirty minutes, and then stopped to recharge the batteries and give ourselves a break.

We let kids operate two of the bots on each side and we operated the third. A member of the museum staff took names of those that wanted to have a go at it and sent the kids to each end at the right time. The next morning, we had a couple more staff to act as "kid wranglers" again and help with crowd control. This worked very well in the morning but as most of the volunteer staff left after lunch, we had some problems in the afternoon in keeping things organized so that everyone got a turn and no one jumped the line. Some parents and children are better behaved than others!

FIGURE 3

The level of play varied a lot with the skill levels of the kids, but they all seemed to have great fun even if they just drove the bots randomly around or attacked the opposition (or even their own side). (See **Figure 3**)

We really needed the breaks every half hour. Chuck's bots were using NiCad batteries and they took the full 30 minutes to recharge. I had plenty of spare A123 packs for my bots, but I was kept busy fixing minor problems. One thing I discovered is that the battery in one of my transmitters could not recharge fast enough so that by the end on the day, it would not last the full demo period. This had not been a problem at combat events where you have one to three minute fights every 20 minutes or so.

I also needed to provide better cooling for the drive ESCs as the combination of more frequent driving and the kids tending to keep full power on all the time (even when up against the wall) was making the controllers shut down due to thermal overload. I had protected the ESC with a little foam as I normally do in a combat event but I had to remove it to keep them working properly for Hockey.

The event ended at 5:00 pm and by that time both me and my bots were just about done! The Museum was very pleased with how popular Bot Hockey was and asked us to bring it back again next year. It would be good to get some local schools involved so that we could have a real competition, as well as demos for the kids.

I need to do some work on my bots to improve their performance. I'll probably fit gyros to improve straight line stability and reduce the sensitivity to steering commands so they're easier for the kids to drive. I'm also going to build at least one more so that I will have a spare to be able to quickly replace any bot that has a problem. The new bot will be custom designed for the sport, and have increased power and traction to stand up better to a game that is about as rough as the NHL version.

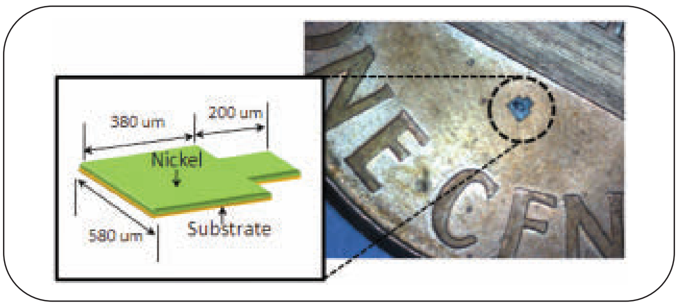
Videos of Bot Hockey can be found on YouTube by searching for "bot hockey."

Bot Hockey will be at the Schiele Museum in Gastonia, North Carolina on 24th of July, 2010. SV

Images courtesy of the Museum of Life and Science in Durham, robot hockey event at Robot Rumble, by the author and Chuck Butler.

Stevens Institute Takes Third at Microrobot Event

A team from Stevens Institute of Technology consisting of undergraduate student Sean Lyttle, along with graduate students Wuming Jing, Xi Chen, and Zhenbo Fu, and led by Professors David Cappelleri, Jan Nazalewicz, and Yong Shi from the Department of Mechanical Engineering, placed third in the Freestyle Competition of the IEEE/National Institute of Standards and Technology (NIST) Mobile Microrobotics Challenge held at the IEEE International Conference on Robotics and Automation in Anchorage, Alaska from May 3-10, 2010. The Stevens team was one of only six teams to qualify for the final round of the Mobile Microrobotics Challenge held at the conference.



The team designed and manufactured a micro-scale magnetostrictive asymmetric thin film bimorph (μMAB) microrobot for the competition. Utilizing the magnetostrictive principle, different bending and blocking forces occur under the arched μMAB due to the in-plane strain generated in the bimorphs by the application of oscillating external magnetic fields in the workspace of the microrobot. The differences in the resulting frictional forces drive the movement of the robot body.

Stevens team at their table at the Mobile Microrobotics Challenge. From left to right: Wuming Jing, Zhenbo Fu, Xi Chen, and Prof. Cappelleri.

Stevens μMAB Microrobot Design

All robots entered in the competition were no bigger than 600 micrometers in their largest dimension and were required to operate without the direct connection of wires, i.e., untethered operation. The competition consisted of three events structured to test each microrobot's speed, agility, and ability to manipulate small objects.

The Two-Millimeter Dash

Microrobots are required to sprint across a distance of two millimeters (beginning from a dead stop) and come to rest in a defined location.

Microassembly

Microrobots must insert pegs into designated holes in a planar assembly framework.

Freestyle Competition

Event to highlight the strengths of their microrobot design by performing a task of the teams' choosing.

Stevens participated in both the Two-Millimeter Dash and Freestyle Competition events at the Challenge. The team's μMAB microrobot had one of the fastest individual runs of the Two-Millimeter Dash event at only 0.027 seconds. The final scoring for the event was the average time across three runs; the team placed 5th out of 11 teams for this event.

In the Freestyle Competition, the team programmed their microrobot to automatically move in both a square and X-shaped pattern inside the 3 mm x 2 mm playing field. Stevens placed third in the Freestyle Competition behind teams from ETH Zurich and Carnegie Mellon University, respectively.

Robot Design Features:

Materials of Construction: Nickel and Copper
Drive Mechanism: Externally Applied Magnetic Fields
Capabilities: Capable of being controlled with both oscillating and gradient magnetic fields. Properly tuned oscillating fields cause vibrations in the robot, resulting in turning or walking. Gradient magnetic fields can be used for steering the robot or for pushing/pulling of the robot for very fast movements.

RoboGames in Retrospect

By Greg Intermaggio

THE TEAM:

Greg Intermaggio
San Rafael, CA

Joseph Brinesh
Kentfield, CA

Jason Halbur
Yorba Linda, CA

THE BOTS:

Disgruntled Chef
Chariot of Fire
Smooth Operator
'Gellan
ATR (All-Terrain Rover)

This year at RoboGames, the competition was bigger and badder than ever before. RoboGames 2010 took place April 23-25 at a new location in San Mateo, CA (about 30 minutes south of Fort Mason, where the event used to take place). Though I've competed many times before at RoboGames, this year was the first in which I was part of a well-organized team, and with five robots, our team Tesla Prime competed in eight events. Here's our story.

This was the pits and our own personal table.

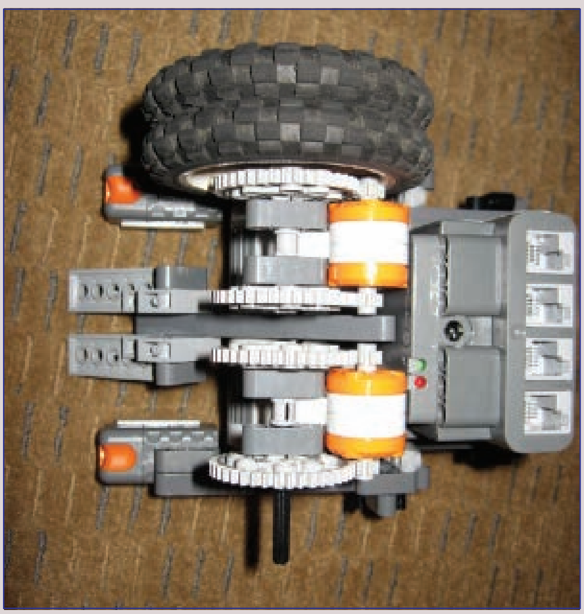
THE EVENTS

Disgruntled Chef ...

ComBots Feathweight - Disgruntled Chef

Our featherweight combot, Disgruntled Chef, is by far the team's most worked on bot. Chef is built from scratch, and together we've spent well over 100 man-hours on his design and fabrication. Despite all that work, there were some design flaws we were aware of from the start that hurt our ability to compete in the tournament. Though the structure was very sound and the critical electronics were very well protected, Chef simply didn't have any way to cause real damage to other robots. We ended up getting flipped upside down and incapacitated in our first round by Shaka, and then lost to Pinball in a judge's decision in our second round — eliminating us from the tournament.

Despite our quick elimination, it was a great learning experience for us, and in many ways our design was very successful. Despite its lack of aggressive capability, Disgruntled Chef was extremely defensive. During those two fights and several exhibition (for fun) matches, we sustained no significant damage. Now all we need is a sledgehammer of doom!



Chariot of Fire

LEGO TubePush - Chariot of Fire

LEGO TubePush is an event in which LEGO robots must move empty toilet paper tubes through a maze and into a finishing zone. Our robot, Chariot of Fire, was originally designed to compete in the LEGO Sumo event*, but due to limited resources (I only have so many LEGO kits!), we used the same chassis to compete in the LEGO TubePush event. Chariot of Fire was programmed the morning of the event due to time constraints, and managed to finish in second place, bringing us home a silver medal!

*Due to a mishap, we were unable to compete in the LEGO Sumo competition this year, so this was one of our four entries that was ready, but didn't make it.

'Gellan

Open Linefollower - 'Gellan

'Gellan was built for fun and as an engineering challenge. We wanted to create a LEGO bot that moved in a unique way by segmenting the front from the back, creating something like a trailer. It was a great design, but unfortunately was a bit too twitchy for the big 90 degree turn on the line following course, and wasn't able to complete its run.

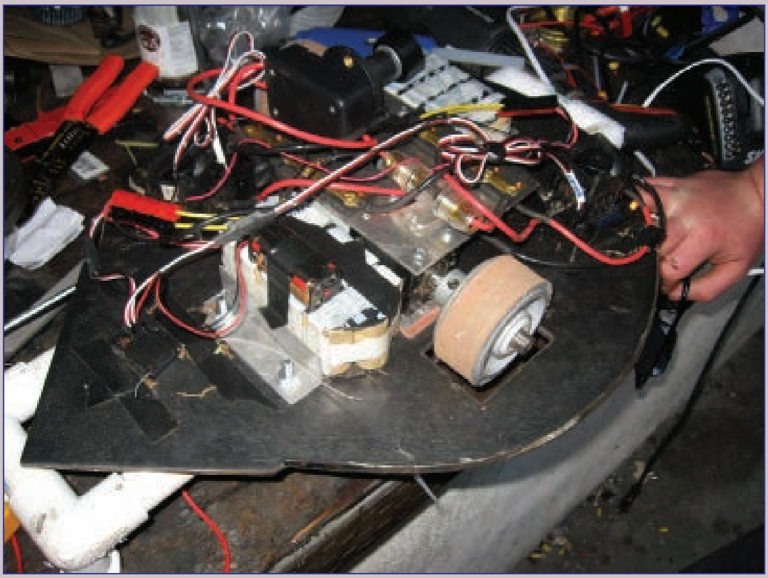
Tabletop Navigation - 'Gellan

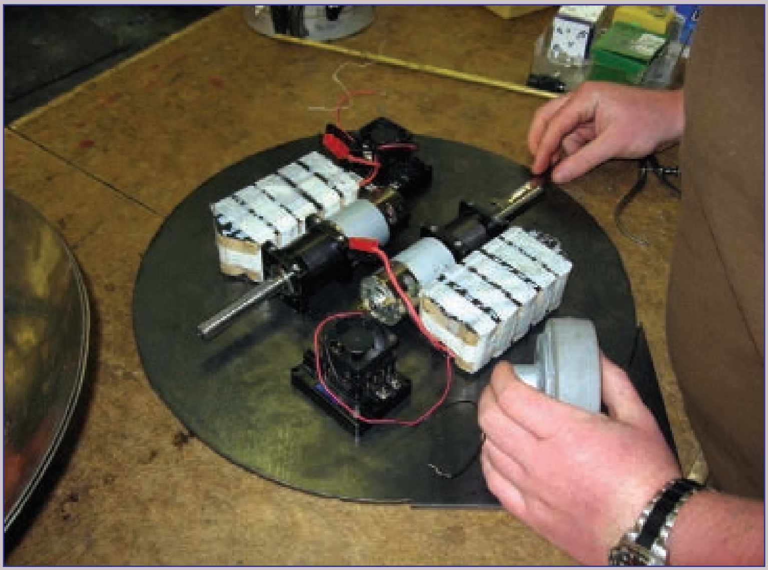
Tabletop Nav is a unique challenge since it's very open-ended and more about creativity than specific performance, unlike other events. The challenge this year was to push a block into a shoe box at the end of a table without falling off the table, and in the coolest way possible. 'Gellan strutted its stuff across the table and into the shoe box, bringing home a bronze for the team.

LEGO Open - ATR

ATR (short for All-Terrain Rover) was a tank-like robot that Jason built to compete in the LEGO Open. It was a very cool design (reminiscent of a packbot), and was capable of traversing over a shoe box which was quite impressive given its size. ATR took home a glorious gold medal for the team which we were all very happy about!

Workin' on the Chef ...





Joe and I have known each other for years, and live a short drive apart. Jason, on the other hand, is a friend from Cal Poly Pomona whom we met the fall of 2009. Being that he lives six hours away from us, Jason had very limited involvement in the development of our main robot — Disgruntled Chef — so he focused instead on our LEGO entries.

After deciding how many events we wanted to compete in — and whether or not we'd actually have a dedicated group of people working on the project — we had about six weeks to go "from bits to bots." At the end of those six weeks was RoboGames, and as it came down to crunch time it became painfully apparent that we had spent too much time at the end of our RoboGames prep, and not enough at the beginning, which brings us to our first lesson ...

Lesson 1: Schedule workshop time!

Firstly, let me say that EVERYONE makes this mistake to some extent at some point. Not scheduling enough workshop time is an easy trap to fall into. It comes down to setting and abiding by your own deadlines, so that instead of one mad rush to get things finished right before game day, you have several smaller rushes many days in advance. It's very easy to misjudge how long any given task will take, and admittedly, we would have benefitted from setting a schedule for ourselves. To put it simply, we didn't spend enough time in the month or so *before* RoboGames that we had to work on our bots, and instead we left lots of work to the last minute. Though it wasn't a fatal mistake, it certainly was a stressful one.

About 1-1/2 weeks before RoboGames, our ComBot "Disgruntled Chef" was drivable, despite needing a lot more work to be combat-ready. At this point, we determined that I would be Chef's driver at the competition. It was wise to make that decision then because we only had one set of batteries, and could only charge and recharge them so many times in the days we had left. This "logic" brought us to Lesson 2.

Lesson 2: Practice!

This is particularly pertinent to ComBots. If you're competing in an event where you have to remotely and precisely control a vehicle moving at high speeds, practice is key. If your team has more than one member, choose a "driver" well ahead of time, and make sure they practice as much as possible. Leading up to RoboGames, I was driving our 30-pound Disgruntled Chef 2-3 times a day most days, in about 15 minute intervals (that's about how long it took to run the batteries down). That driving practice really showed — and we'll certainly build on that success at our next competition.

You never realize how much time and effort it takes to build a robot from scratch. Even just a simple curved bracket has to be measured, cut to size, marked, center punched, drilled out, bent into a precise curve, re-drilled to ensure the hole didn't shrink, and tapped so it will thread. It's very easy to overlook the details of a specific component and greatly misjudge how long it will take to fabricate. Since every little bit takes so much work, it's very important that you delegate.

Lesson 3: Delegate!

The delegation of work is a key to success beyond the world of robots — in whatever you do, it's important to make sure that when there's work to do, everybody is contributing as much as possible. One of the great things about working with friends in a team environment is that since you're all working towards a big success, you get to boss each other around! When we were in the machine shop, I was king. When we were at Joe's home shop, he was lord of the land, and when we were working with LEGOs, Jason was the guy to answer to. All three of us understood the arrangement, and because of it we were able to delegate tasks effectively.

This ability to delegate helped us immensely — both before and during the competition. Joe was in charge of catering for the RoboGames Builder's Party, so was occupied by that most of the competition. This left me in charge of the ComBot, and Jason in charge of the LEGO entries. I can't stress enough how much delegation helped us, and how much potential it has to make or break a team. In fact, arguably our biggest failures at RoboGames were because we didn't delegate.

Lesson 4: Know the event!

One of those little things that is easy to keep putting off, but ultimately somebody needs to take responsibility for is knowing the event and distributing that information to the team members. In our case, we made the assumption that one glance at the event schedule would be all the information we needed — this was very far from reality. It was a mad rush at the competition for us to get set up at two different pit tables (one for ComBots and one for everything else), and get our bots checked in.

It was also assumed by everyone on our team (myself included) that I knew everything we needed to know about competing. The fact was we had four entries that were 100% operational that didn't actually compete because they didn't get checked in at the right time, or nobody took them to the area of the competition. It was mostly a case of too much to do in too little time. We should have planned ahead to delegate these responsibilities at the event. Having only five of nine entries competing certainly frustrated the team a bit, and added to the stress of the event. Despite that frustration, we kept on trucking because we realized that there were better things to do than argue and complain.

Lesson 5: Accept the ref's decision!

During one fight of our ComBot, I was certain I won. I outdrove, outpowered, and got underneath my opponent to drive him across the arena. When the judges turned in their decision, though, I had lost. The thought briefly flashed into my mind that I should dispute the decision — I was so sure I had won! I realized, though, that arguing would be unproductive, and so I contained my frustration and accepted the loss with good sportsmanship.

During RoboGames 2010, we met lots of people with lots of ideas about robots. Some of the ideas were awesome, and some of them less so, but all of them valid and interesting. We had some very stimulating conversations with some very interesting people, and learned a lot.

Lesson 6: Learn from everything!

This is another one of those ideas that can help you in more than just robotics — you can (and should!) learn from everything. Not only should you gain insight and knowledge from your accomplishments, you should also learn from your mistakes — and the accomplishments and mistakes of others. We got some great ideas for future robots, and shared some of our own, as well!

Wrapping Up

Six weeks of building, programming, and tinkering later, all our hard work came to culmination at the world's biggest robotics competition. We brought three team members, four robots, and five entries to RoboGames 2010, and took home an unforgettable experience.

After all was said and done, Tesla Prime took home medals in three different events, bringing home a total of one gold, one silver, and one bronze. Everyone learned a lot, and you can bet we'll be back next year with bigger, badder, and bolder entries for RoboGames 2011! SV

It's amazing what you can stuff in a trunk for a road tip to RoboGames ...

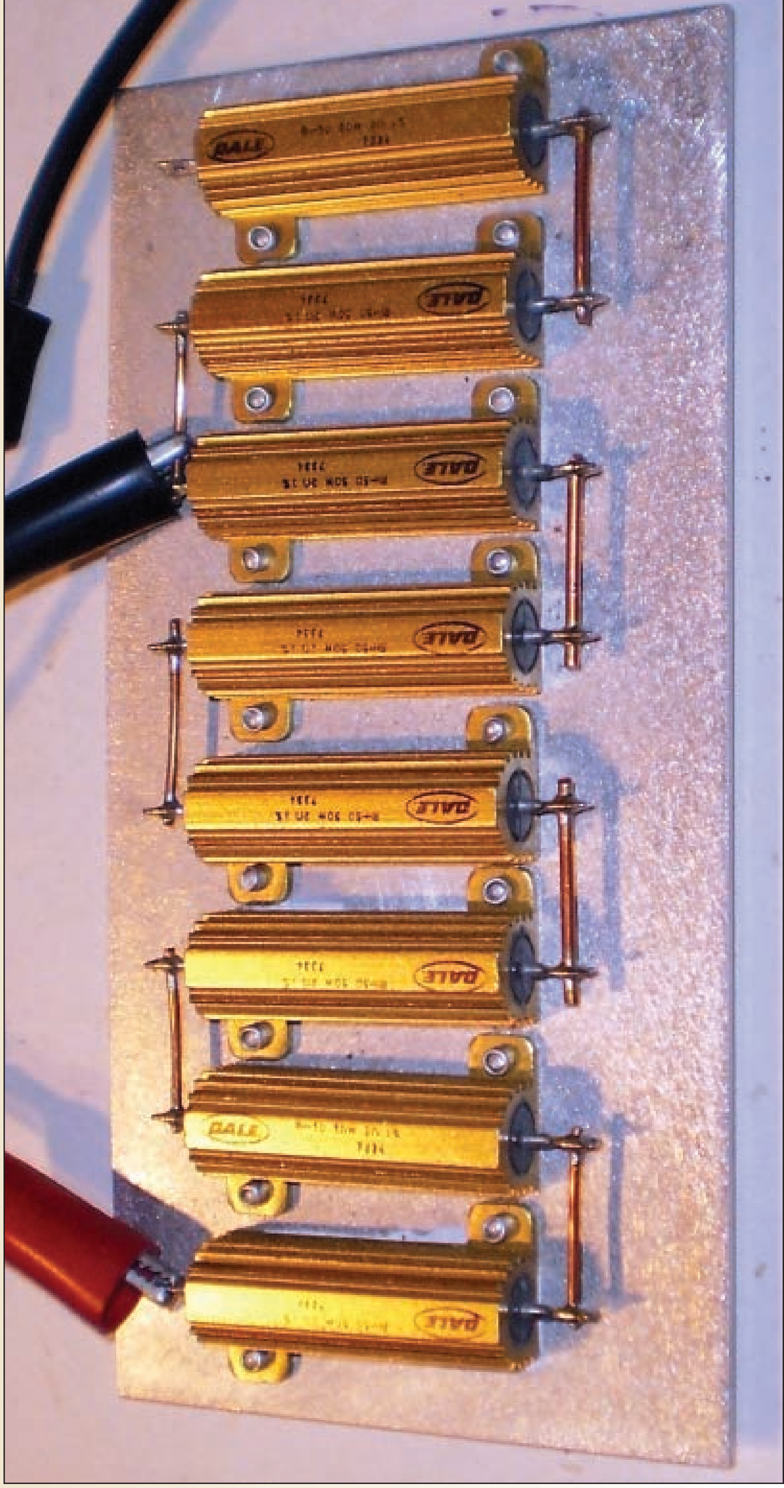


FIGURE 1. Load resistors and heatsink.

Determining a Rechargeable Battery's mAh Capacity

By Clark Robbins

Many of us use cordless tools when building or repairing our robots that have NiCad or NiMH rechargeable batteries, but rarely do we know what the battery's actual mAh capacity is. (For some reason, manufacturers do not include this information.) I was facing this problem. I had lots of power tool batteries (over 20, mainly 12V and 18V) but they didn't seem to hold a charge very long or have much capacity. To determine the condition of these batteries, I set up a relatively simple test system to measure a battery's mAh capacity using a PC, some load resistors, a serial port A/D, and some software (VC#). My latest version uses a couple of relays and more A/D channels to test two batteries simultaneously.

Take Charge!

The easiest way to determine a battery's capacity is to fully charge it, then discharge it at its mAh rating until the cell voltage equals 1.0 volts. The mAh rating = (Average Current) * (Time it takes to reach a cell voltage of 1.0 volts). If you don't know the battery's mAh rating, then just use an estimate for the discharge current. Most power tool battery packs range from 1,200 mAh to 2,000 mAh. I used a load that would average approximately one amp. The actual rate used would only be critical if you were manufacturing batteries. For our purposes, one amp will work just fine. For AA, C, and D cell rechargeables, however, I would recommend 0.25 to 0.4 amps. Keep in mind that your load resistors will be dissipating a lot of power — more than 20 watts in some cases — so you will need to heatsink them or use a fan to provide cooling. I used eight Dale 50W two ohm aluminum housing power resistors (in series to achieve 16 or 12 ohms). I mounted them on a large aluminum plate using heatsink compound between the resistor and the plate.

Here's an mAh calculation example: Estimated Average Current (I) = [(Initial Battery Voltage + Ending Battery Voltage) / 2] / Load Resistance. If IBV = 18, EBV = 15, Load Resistance = 16 ohms, and Time = 1.5 hours, then the Average Current (I) is 1.03125 amps and the mAh capacity is 1,547 (Average Current (I) * Time). The actual average current will be slightly different due to the non-linearity of the battery voltage during discharge (see curves shown in the **figures**). For our purposes, this calculation will be sufficient to get a good estimation of the battery's mAh capacity.

In the enhanced test setup, I measure the actual current using an additional A/D (Analog-to-Digital) channel. You really don't need to do this as the extra

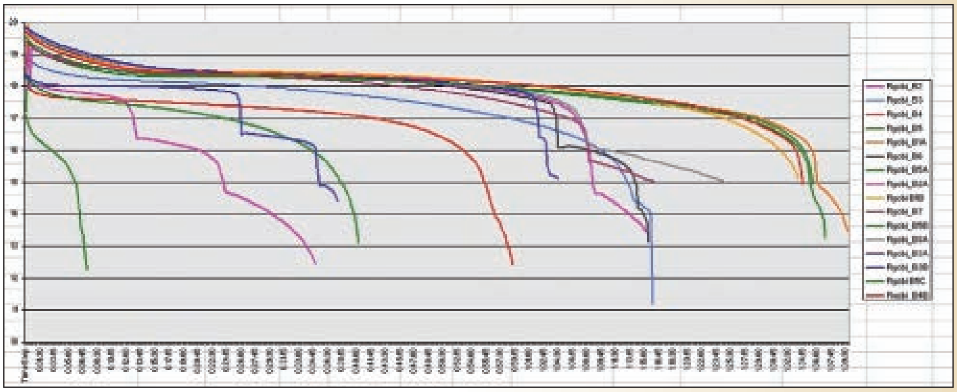


FIGURE 2. Excel plots of battery discharge curves.

B	C	D	E	F	G	H	I	J	K	L	M
	Test Date	Status	InitBV	EndBattV	TVDrop	PVDrop	OneVoltCellTime	VPerMin	MinsPer1VDrop	AvgCurrent	MAH
Ryobi 18v Pack #B1	12/4/2004	Charge	18.184	14.101	4.083	0.135	7.50	0.544	1.837	1.364	170.50
Ryobi 18v Pack #B1A	12/4/2004	Very Good	19.803	14.981	4.821	0.689	96.50	0.050	20.015	1.127	1812.69
Ryobi 18v Pack #B1B	9/23/2006	Very Good	18.667	15.120	3.547	0.167	94.06	0.038	26.518	1.056	1655.21
Ryobi 18v Pack #B2	12/4/2004	Charge	18.297	15.080	3.217	0.245	24.25	0.133	7.539	1.064	430.03
Ryobi 18v Pack #B2A	12/4/2004	Good	19.294	14.915	4.379	0.606	69.25	0.063	15.814	1.136	1311.13
~~Ryobi 18v Pack #B2B~~	~~9/23/2006~~	Bad	8.600	Battery Pack Failed							0.00
Ryobi 18v Pack #B3	12/4/2004	Good	18.996	15.000	3.996	0.132	73.50	0.054	18.399	1.099	1346.28
Ryobi 18v Pack #B3A	9/26/2006	Charge	18.373	14.890	3.483	0.551	35.75	0.097	10.265	1.124	669.46
Ryobi 18v Pack #B3B	9/26/2006	Good	19.919	14.983	4.936	0.14	65.25	0.076	13.218	1.179	1282.17
Ryobi 18v Pack #B4	12/4/2004	Fair	18.039	15.036	3.003	0.183	56.00	0.054	18.646	1.073	1001.47
Ryobi 18v Pack #B4A	9/26/2006	Charge	16.483	14.226	2.257	1.032	1.50	1.505	0.665	1.007	25.18
Ryobi 18v Pack #B4B	9/26/2006	Very Good	19.717	14.926	4.791	0.474	94.50	0.051	19.725	1.161	1828.33
Ryobi 18v Pack #B5	12/4/2004	Charge	17.145	15.052	2.092	0.156	6.25	0.335	2.987	1.002	104.38
Ryobi 18v Pack #B5A	12/4/2004	Very Good	19.435	14.903	4.532	0.422	95.75	0.047	21.128	1.119	1785.74
Ryobi 18v Pack #B5B	9/26/2006	Charge	18.285	14.957	3.328	0.108	38.25	0.087	11.494	1.095	698.10
Ryobi 18v Pack #B5C	9/26/2006	Very Good	19.556	14.919	4.637	0.507	95.75	0.048	20.649	1.154	1841.59
Ryobi 18v Pack #B6	12/4/2004	Good	19.851	14.938	4.913	0.245	74.50	0.066	15.164	1.171	1453.40
Ryobi 18v Pack #B7	9/23/2006	Good	19.417	15.000	4.417	0.029	76.40	0.058	17.297	1.076	1369.51
Ryobi 18v Pack #B8	9/25/2006	Charge	16.331	14.999	1.332	0.022	9.75	0.137	7.320	0.979	159.10
Ryobi 18v Pack #B8A	9/26/2006	Very Good	19.727	14.989	4.738	0.041	85.00	0.056	17.940	1.112	1658.89

InitBV = Initial Batt Voltage at Start of Test
NowBattV = Batt Voltage at this Test Point
TVDrop = Total Voltage Drop from Start of Test to This Point (InitBV - NowBattV)
PVDrop = Voltage Drop for this Test Period (Previous NowBattV - This NowBattV)
OneVoltCellTime = Minutes Elapsed from Start of Test to This Point (1.0 Volt Point)
VPerMin = Voltage Drop per Min
MinsPer1VDrop=Minutes for a 1 Volt Drop
AvgCurrent = (InitBv / Resistance) + (EndBattV / Resistance) / 2

FIGURE 3. Actual data in Excel used to create discharge curve plots.

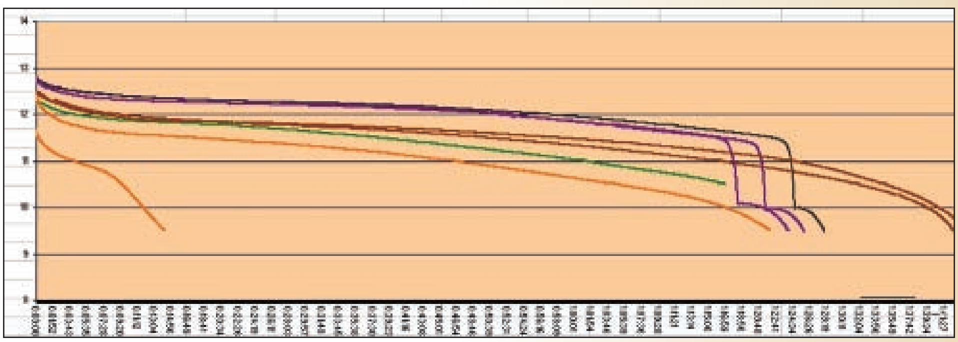


FIGURE 4. Battery discharge curves showing comparison between NiCad and NiMH batteries.

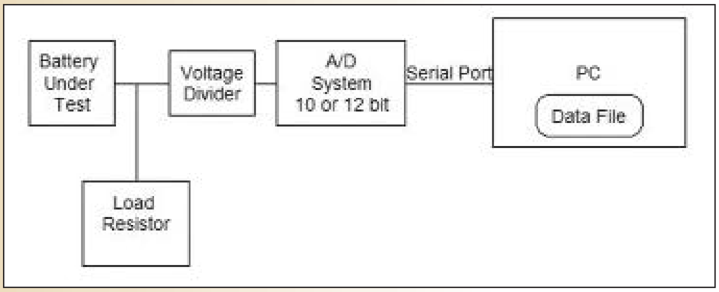


FIGURE 5. Block diagram of the test system.

accuracy is of academic value only.

Look at the accompanying plots (in Excel) of a number of tests that I ran on eight Ryobi 18 volt battery packs. You can see that quite a few of the packs had very low mAh capacity. However, if you compare by colors (same battery pack), you can see that some of the tests also had acceptable results. The reason is that a full charge/discharge(test)/charge/discharge(test) can sometimes restore a NiCad battery's capacity to near normal. If it doesn't, then that pack is probably defective. You can also tell by the plots whether a pack has any failed cells by a step voltage drop early in the discharge cycle.

Figure 3 shows the actual data used to create the plots in Excel. If you look at it closely, you can see that the Charge/Discharge (test) cycles resulted in some packs more than doubling in mAh capacity (see tests B5B and B5C).

If you have any battery packs that are NiMH, you will see a big difference in the discharge curves. The NiMH will have a higher and more constant voltage during discharge, but have a big step drop at the end when it reaches its capacity end. The three dark colored discharge curves (**Figure 4**) are NiMH battery packs.

One thing I have noticed is that over time, both NiCad and NiMH battery packs will lose mAh capacity that cannot be recovered even by discharge/charge cycles. I am researching whether there is really any viable means to recover any of that lost capacity. I believe that the fast charge type of battery chargers may, in fact, contribute to this eventual loss of capacity.

FIGURE 6. Test system electronics' housing and connections.

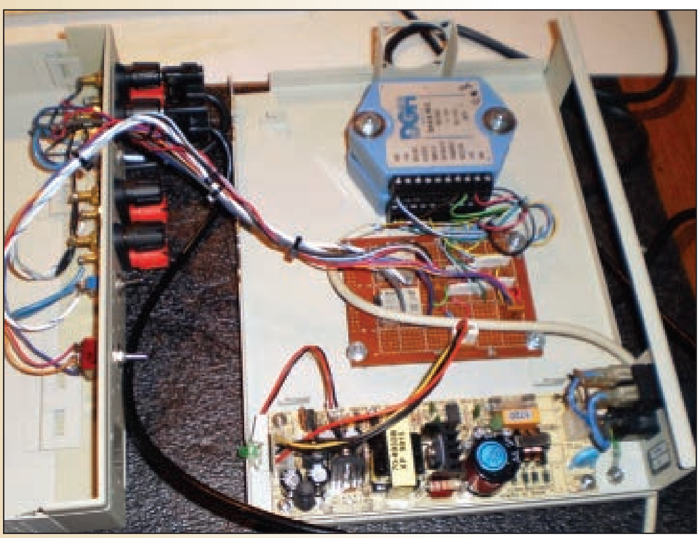


FIGURE 7. The electronics consisting of the A/D modules, control circuit, and power supply.

Hardware Design

Check out the block diagram of the original test setup in **Figure 5**. To make the connection to the battery easier, I modified each battery charger by adding terminals that connected to the charger's battery terminals. *Make sure that you understand that with some battery chargers there can be lethal voltages at these points if plugged into 120 VAC with no battery pack installed!!! NEVER PLUG IN THE BATTERY CHARGER WHILE TESTING OR PREPARING TO TEST A BATTERY PACK!*

I have used DGH and Advantech serial A/D modules but any A/D that you can easily interface (serial A/D, serial DMM, Ethernet, USB) to your PC will work. The software provided is for example only and you will have to modify it (within VC#) to work with whatever A/D system you are using. A/D modules with a PC interface are readily available for a reasonable cost. With some persistence and patience, you can find a useable A/D module at even less cost (auction, closeouts, or using a microcomputer chip with A/D).

In terms of actual hardware, there are no real critical components. However, for the voltage divider you should match values as closely as

possible. You can compensate for any errors in the voltage divider by using a calibration factor in the test program. The load resistor value should be within range of the estimated mAh capacity and be able to handle and dissipate the power. You can either just connect the battery to the load resistor with a jumper lead or use a digital I/O line (from the A/D module) to drive a relay to handle the connection. The relay should be spec'd to handle the actual load current plus a safety margin (100%), and the relay driver can be just about any Darlington transistor with enough gain to drive the relay and a current rating to handle the relay.

If you add current sensing, the current sense resistor must be low enough to minimize the effect on the load, yet large enough for the A/D to measure. One advantage of the Advantech module is that each channel's gain is programmable. You can measure the voltage with one channel and the current with another channel.

If you do not use a software-controlled relay to make/break the connection to the load resistor, you will have to monitor the battery voltage and disconnect the load when the battery voltage drops to one volt per cell. Discharging much beyond that point is not recommended as it can degrade a battery's performance.

I am also considering building a completely stand-alone microcomputer-based battery tester/charger. You could take the basic design described here but use a microcomputer (chip or board system) with A/D and digital outputs, just interfaced to an LCD display. It would be best to use a system with a high level language, as the math involved would be much easier to program that way. The tester could be integrated into a standard battery charger, but with relays to separate the charge and test functions. For our PC-based system, I charge the batteries using a separate charger unit. That way, I can test one battery while recharging another.

The Software

The software was written in Microsoft Visual C# 2008 Express Edition which is available for free from Microsoft.

When the Start Battery Test button is hit, the software initializes the serial port, creates the Output Data file, turns on the load relay, initializes a one second timer, and begins taking measurements. In the early setup, I would just connect the load with jumper clips, then

FIGURE 8. Access points to the battery terminals.

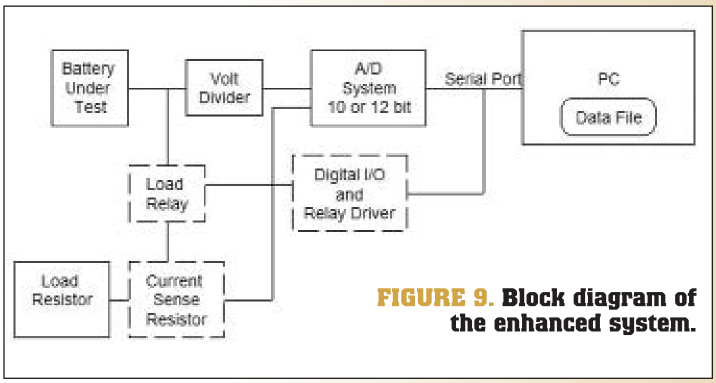


FIGURE 9. Block diagram of the enhanced system.

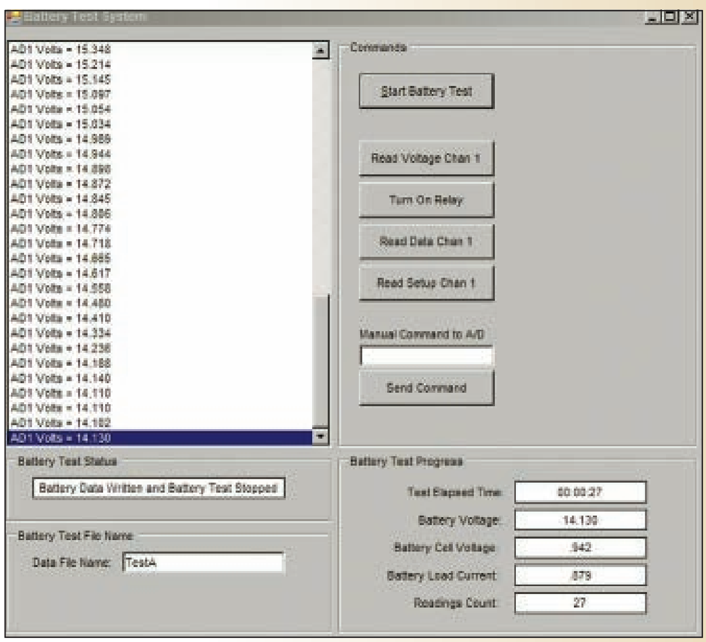


FIGURE 10. Test system software control panel.

hit Start. The latest version uses relays to connect the battery pack to the load that is activated by the program when the Start Battery Test button is hit. The program then makes various calculations and saves each reading to the Output Data file. The final calculation is the mAh rating which is the last line in the file.

```
// we compute the following;
// - total voltage drop from first batt
// value
// to last batt value (at cell=1v)
// - MAH by average current/elapsedtime
double dElapsedTime = Convert.ToDouble
(iTimeToCellOneVoltageDrop);
double MAH = (dAvgCurrentToCellOneV *
((dElapsedTime) / 3600)) * 1000;
// avg current (in amps) * time (in hours)
// = amp hours * 1000 = MAH (millamp hours)
```

The program measures the time it takes for the battery pack to reach a cell voltage of 1.0 volts. It also measures the average current for this same time period. It then computes the mAh rating by multiplying the Average Current * TimeElapsed (in hours).

The program saves the calculated data into a CSV data file that is easily imported into Excel. It would not be difficult to create an actual Excel data file, but that's for a future enhancement. It is also not necessary to use Excel since the last line in the data file contains the summary data, including the calculated mAh capacity of the battery tested.

Final Check

I hope this gives you some ideas on how to test and salvage all those rechargeable batteries that you have lying around. I spent less than $100 on everything and have saved probably more that triple that amount by restoring many of the batteries that were not performing. SV

DGH and Advantech A/D modules are available at **www.dghcorp.com/**

www.advantech.com/products/Remote-I-O-Modules/sub_1-2MLCJF.aspx

I bought all of my A/D modules used for a fraction of the new price. I also have used an inexpensive DMM w/RS-232.

I check eBay (**www.ebay.com**), Craigslist (**http://detroit.craigslist.org/**), and also use Google Shopping (**www.google.com/products**). You just have to be patient since they don't show up very often.

How to do this project for peanuts!

I have saved almost $300 using my battery testing setup. A typical cordless tool battery costs $30-$40 each, with some costing $50-$60 each. The methods and equipment I chose will do the job of battery testing quite well, *but you can accomplish the same for possibly under $30.*

Let's start with the load resistors. You can use high power surplus resistors if the values result in approximately 12 to 24 ohms (for 12V to 24V batteries) and are capable of handling 20-30 watts. It's even cheaper to use automotive bulbs. Two 1156 bulbs in series is 12.2 ohms (hot) and they are under a $1 each. If you want to save more money and not use sockets, just solder the wires to each bulb, but make sure they do not touch anything as they get very hot. I recommend using a relay to turn the load resistor on and off. If you don't have an output to control the relay, then just wire a switch in series with the resistors or bulbs. Then when you are ready to begin, flip the switch on and start your testing.

The other major component is the A/D system. If you have a DMM with an RS-232 output, then that is all you will need. Connect the RS-232 output to your PC. Then, modify the software to work with the DMM.

Another option is an A/D demo kit that has an RS-232 output. I have seen some of these for as low as $20. Many manufacturers offer these demo/evaluation kits or boards to showcase a particular A/D chip. For battery testing, even an eight-bit A/D would suffice.

Another option is a cheap PIC kit with A/D. Many of these have an RS-232 interface that can be used to program the chip, and then can be used as an interface to a PC.

You could even dispense with using a PC and build a completely stand-alone battery testing system. Just use an inexpensive micro with A/D and an LCD display. I have seen some Stamp modules with A/D for $20 and an LCD for under $10. If you shop around, you could spend as little as $30 for the entire battery test system as either a stand-alone or one with a PC interface.

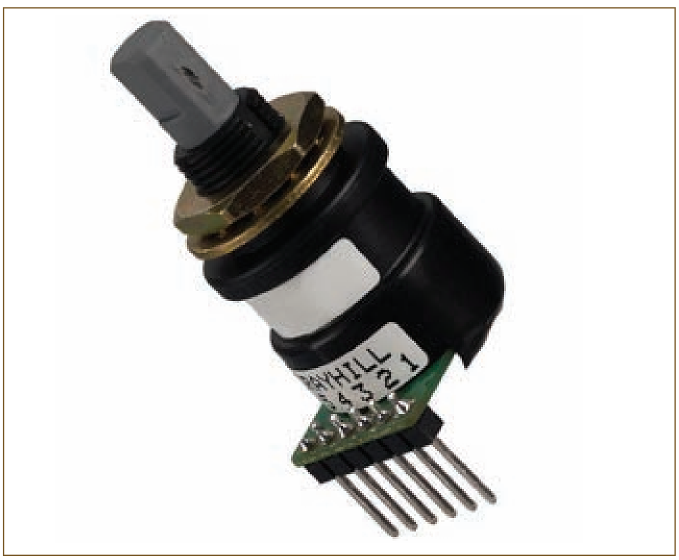
It will take some programming changes but all the principles are in the software program available from the *SERVO* website (**www.servomagazine.com**). Basically, all you have to do is measure the starting voltage and the ending voltage (when each cell = 1V) and the time elapsed. Then, based on your load resistance, compute the average current using the starting current and ending current. Multiply the current times the elapsed time and you have the battery's mAh capacity. Even using an eight-bit A/D and measuring the voltage every 30 seconds will still come up with a usable mAh calculation.

See How Easy Coding A Rotary Optical Encoder Can Be

By Fred Eady

I can remember back when the only way to interface with a computer was with a piece of hard paper called a punch card. Depending on the company you kept, punch cards were also known as IBM cards or Hollerith cards. Time passed and punch cards gave way to terminals. Again, depending on the parties you attended, a terminal was an ASCII device or a 370 device. Computing devices offered by DEC used VT-100 terminals while the IBM mainframe crowd interfaced to their machines with clusters of 370 terminals. In the beginning, terminals were monochrome devices and as time marched on, the green and gray screens became colorful as DEC and IBM offered newer and fancier color versions of the VT and 370 human interface devices.

PHOTO 1. This particular optical encoder mounts in a standard 3/8 inch hole, has 32 detented positions, and an SPST normally open at the bottom of its shaft.



Early computers aimed at the home and small business market also used various types of terminals as human input devices. In the late 1980s, IBM finally decided that ASCII wasn't a dirty word and produced the very popular 3151 series of ASCII terminals. In fact, I have a 3151 in the closet and even today we still use a terminal of sorts, which is part of today's desktop PCs.

ASCII terminals found their way to embedded platforms via the RS-232 protocol. Since ASCII terminals were not a device that could be found in every household, PCs equipped with terminal emulation software and RS-232 ports eventually annexed a huge amount of ASCII terminal territory.

As embedded devices became smaller and more portable, the ASCII terminal and PC terminal emulator no longer fit the paradigm. Pushbuttons and LED displays became the human interface components of choice for the embedded designer. LEDs gave way to LCD devices and discrete pushbuttons morphed into keypads. Today's cell phones are good examples of how far the LCD/keypad combination has come.

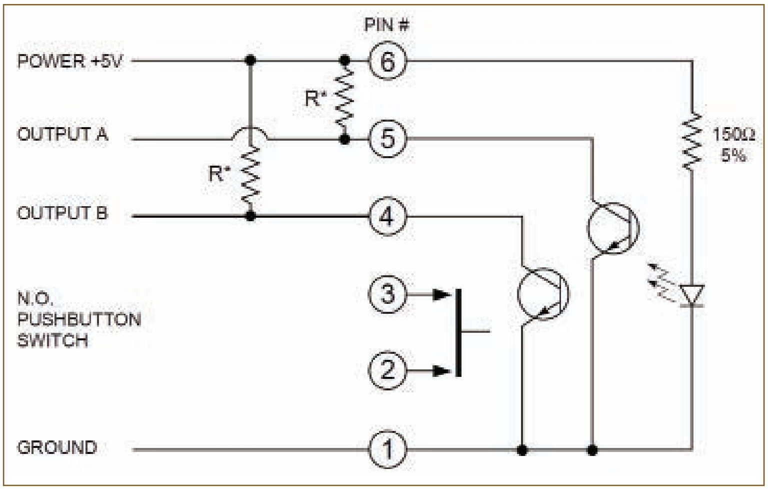
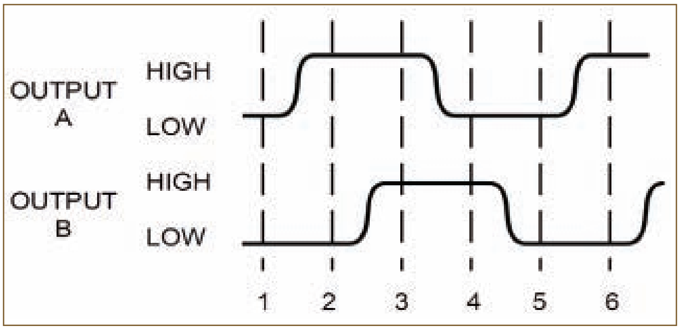


FIGURE 1. This is a simplified schematic of the Grayhill Series 61C optical encoder. The selection of the resistor values determines the logic level of the optical encoder outputs. Our design is powered by five volts. So, the recommended TTL pullup resistor value is 8.2 KW.

Sometimes a keypad is not the most practical human I/O device for an embedded design. For instance, a CNC machine controller may not have room for a full QWERTY keyboard layout. Also, a QWERTY keyboard layout would be overkill for setting the position of a cursor or selecting a simple RPM of drill depth value. In the case of the CNC example, a rotary optical encoder with an integral pushbutton switch is the most logical choice.

As a roboteer, you're used to seeing optical encoders at the end of a motor shaft. Rotary optical encoders are also awesome human input devices. If you've been avoiding them in your robotic controller data entry gear because of the complicated code it takes to support them, cast aside your fears as we're about to assemble and code a universal rotary optical encoder input device.

FIGURE 2. There are multitudes of ways to interpret the phase relationships and logical relationships of this quadrature signal pair. I like to keep things simple. So, instead of writing yet another optical encoder driver, we'll use some specialized hardware to do the job.



Optical Encoder 101

A typical optical encoder employs a light mask, an LED emitter, and a pair of phototransistors to generate a quadrature two-bit signal. The optical encoder we will use in our design is a Grayhill Series 61C device like the one you see in **Photo 1**. The part number of our optical encoder is 61C11-01-00-02 and it can be had from Digi-Key. You already know what the 61C in the optical encoder part number represents. The pair of digits that follow 61C tell us that this optical encoder detents every 11.25° which equates to 32 positions per rotation. Quadrature output coding is denoted by the *01* digit pair in the optical encoder part number. The *00* digit pair is associated with optical encoders that have no detents. Since our optical encoder contains detents, these digits are "don't cares." The presence of a pushbutton switch is announced by the final two digits of the optical encoder part number. The optical encoder in our possession is not designed to be at the end of a motor shaft. It was specifically designed as a data input device.

A highly simplified schematic diagram of the 61C11-01-00-02 optical encoder can be seen in **Figure 1**. The quadrature output signals that emit from the 61C optical encoder's outputs are laid out in **Figure 2**. Note that from left to right, the A output waveform leads the B output waveform by 90°. Just for kicks, let's call left to right forward rotation. Now, if we follow the waveforms from right to left, output B leads output A by 90°.

According to our forward designation, output B leading output A is a sign that the rotation is reversed. Note also that the **Figure 1** logic sequence repeats every four positions. The leading edges and/or the output A and output B logical states at each position (every 90°) can be used to determine the direction of rotation and the number of positions that have been traversed.

Coding with Hardware

You're probably expecting to see some fancy quadrature encoder interface code. Instead of decoding the optical encoder outputs with mnemonics, we're going to use a specialized piece of hardware called a QEI (Quadrature Encoder Interface).

Detent is the term for a method — as well as the actual device — used to mechanically resist or arrest the rotation of a wheel, axle, or spindle.

A detent can be used to intentionally divide a rotation into discrete increments or, as in perhaps its original concept and most rudimentary form, to simply arrest rotation in one direction.

Courtesy of www.answers.com

QEI hardware can be found onboard all variants of the PIC18Fxx31 microcontrollers. The PIC-based QEI hardware decodes motion sensor information provided by the optical encoder outputs. The QEI can also be configured to sense rotational velocity. In that we're not spinning our optical encoder with a motor shaft, we won't cover the QEI velocity configuration in this discussion. We'll base our project here on the 28-pin PIC18F2431. If your optical encoder application requires a bit more I/O, you can use the PIC18F2431 optical encoder logic and code we're about to discuss with the 40-pin PIC18F4431.

The PIC18F2431's QEI consists of three inputs. In addition to the QEA and QEB quadrature inputs, an index signal input (INDX) is also provided. The index input is used with encoders that emit an index pulse to mark an absolute position of rotation. As you can see in **Figure 1**, our optical encoder does not provide an index signal.

Registers and buffers dedicated to the QEI keep track of the optical encoder's direction of rotation and quantity of optical encoder sequences. The UP/DOWN bit in the QEICON (Quadrature Encoder Interface Control) register is used to monitor the optical encoder's direction of rotation. A 16-bit up/down counter keeps track of the rotational position. We'll use the POSCNT register pair to generate a numeric entry according to the value of the POSCNT register. The maximum value of the POSCNT counter is controlled by the value contained within the MAXCNT register.

For instance, if we only want to count from 0 through 9, we load the MAXCNT register with 9. We then configure the QEI to reset on a match of the POSCNT and MAXCNT values. Thus, as we spin the optical encoder shaft the POSCNT will increment or decrement between the values of 0 and 9. If the value is to roll off of 0 during a decrement operation, the POSCNT counter is loaded with the MAXCNT value. Otherwise, the QEI will sense the optical encoder outputs and count from 0 to 9, then rollover to 0.

The PIC C compiler of choice for this project is HI-TECH PICC-18 PRO. Let's use the HI-TECH PICC-18 PRO tool to configure the PIC18F2431's QEI:

```
//************************************************
//*    CONFIGURE QEI
//************************************************
       QEICON = 0b10011000;
       MAXCNTH= 0;
       MAXCNTL        = 9;
       POSCNTH        = 0;
       POSCNTL        = 0;
```

Working the QEICON register bits from left to right, setting the most significant bit of the QEICON register disables the QEI's velocity mode. We really don't care how fast we're turning the optical encoder shaft as our intent is data entry not motor control. The cool thing is that the QEI hardware implementation is very fast and we would really have to try hard to outrun the POSCNT counter rotating the optical encoder shaft by hand.

The zero in the bit 6 position of the QEICON register is an error bit which does not apply unless we configure the QEI for index mode. The QEI Mode bits are located between bits 4 and 2 of the QEICON. QEICON bit 5 is read only and is set when the direction of rotation is forward.

The *110* bit pattern in the QEI Mode bits area signifies that the QEI is enabled in 4x Update Mode, and the position counter will reset on a period match of POSCNT and MAXCNT. There are two QEI Update Modes that can be configured: 2x Update Mode and 4x Update Mode. The QEI 4x Update Mode is a higher rotor position resolution configuration since in this mode the counter increments and decrements on each QEA/QEB input pulse pair. In 4x Update Mode, the position counter (POSCNT) is clocked on every QEA and QEB edge. The 2x Update Mode clocks the position counter on the QEA edge input only. From a user perspective, the 4x Mode operation differs from the 2x Mode operation in that 2x requires a physical movement of two detents versus a single detent movement in 4x Mode to clock the position counter. The final least significant bits of the QEICON register are related to Velocity Mode and are not used in our application.

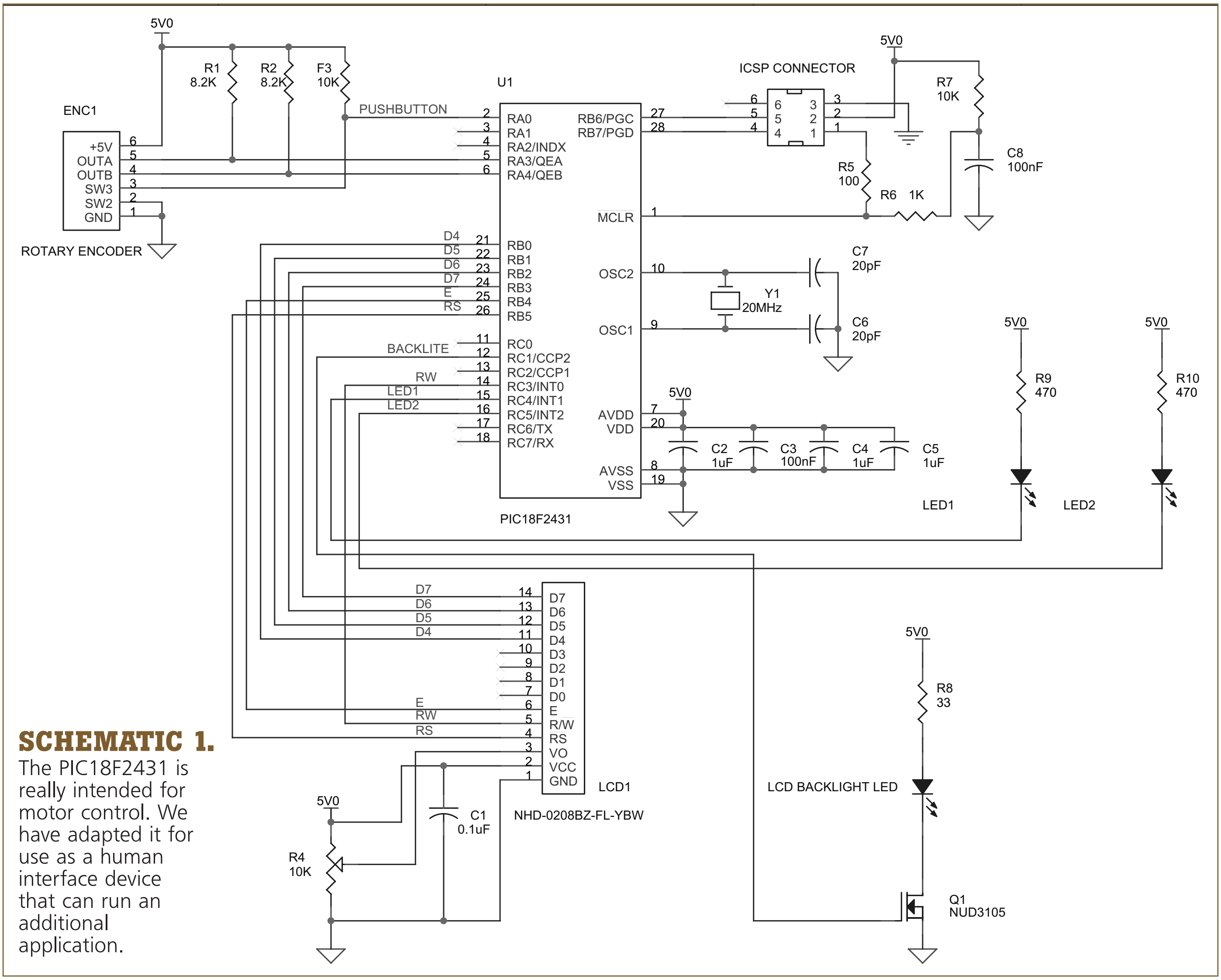
The POSCNT counter is 16 bits wide, so it stands to reason that the MAXCNT register would follow suit. Our QEI configuration code limits the POSCNT to values between 0 and 9 by loading MAXCNT with 0x0009.

Believe it or not, that's it. Every time we spin the optical encoder shaft through a detent, the POSCNT will increment or decrement depending on the direction of the optical encoder's shaft rotation. All we have to do is use the POSCNT value to our advantage. Don't forget. We also have the optical encoder's pushbutton at our disposal.

Adding an LCD

We can twist the optical encoder's shaft until the cows come home and never be the wiser of the optical encoder's operational output. So, let's add a visual output device to our optical encoder input design. The LCD/optical encoder/PIC18F2431 circuitry is graphically displayed as **Schematic 1**. The QEI is pinned out on the PIC18F2431's PORTA. The optical encoder's pushbutton is logically part of the QEI. So, I selected RA0 for the optical encoder's SPST switch input. That leaves RA1 for any analog-to-digital converter (ADC) work you might want to perform. I've also intentionally left the RA2 pin open as you may need the INDX functionality in your application.

Normally, the most significant nibble of the LCD data I/O is connected to the most significant nibble of an eight-bit PIC I/O port when four-bit LCD mode is employed. The



SCHEMATIC 1. The PIC18F2431 is really intended for motor control. We have adapted it for use as a human interface device that can run an additional application.

high nibble of PORTB was not available as I wanted to include a programmer/debugger portal in this design. I also considered losing the UART if I tied the high nibble of the LCD I/O to the high nibble of PORTC. So, I crammed as much of the LCD interface onto the PORTB I/O pins.

If you don't have any plans for reading the LCD's internal SRAM, you can reclaim RC3. Just be sure to wire the LCD's R/W pin to ground. Don't need a back light on you LCD? If not, you can recover RC1 too. The LEDs are here for example. That leaves all of PORTC for your optical encoder design. If you're really pushed for I/O, you can also recover the MCLR pin as RE3.

The C2-C5 bypass capacitor arrangement isn't very clear in **Schematic 1**. Capacitors C2 and C3 are physically attached between the AVDD and AVSS pins, while C4 and C5 lie in parallel between the VDD and VSS pins.

The LCD back light requires a maximum of 80 mA of current depending on the current-limiting resistor value you choose. That's way too much current for a PIC18F2431 I/O pin to absorb. So, I opted to drive the LCD back light with a MOSFET. Doing this allows the PIC18F2431 to control the LCD back light and eliminates the need to worry about the relatively high LCD back light current requirement. I wrote a four-bit LCD driver that consists of an LCD init function, a character/command write function, and a cursor locate function. Here are the base LCD definitions and macros:

```
//************************************************
//*       LCD DEFINITIONS
//************************************************
#define databus              LATB
#define lcdcntrl             LATB
#define E                    0x10
#define RS                   0x20
#define RW                   LATC3
#define clrRW                RW = 0
#define setRW                RW = 1
```

```
#define clrRS                  lcdcntrl &= ~RS
#define setRS                  lcdcntrl |= RS
#define clrE                   lcdcntrl &= ~E
#define setE                   lcdcntrl |= E

#define lcdcls
lcd_send_byte(0,0x01)
#define line1                  lcd_gotoxy(1,1)
#define line2                  lcd_gotoxy(2,1)

#define backliteON             LATC1 = 1
#define backliteOFF            LATC1 = 0

char LCD_INIT_STRING[5] = {0x28,0x08,0x01,0x06,
                           0x0E};
```

The LCD definitions and macros are pretty much self-explanatory. The hex bytes that associate with E and RS represent bit locations within PORTB. The PORTB bit positions are used by the set and clear macros.

All of the four-bit LCD driver functions are based on the *lcd_send_nibble* function:

```
void lcd_send_nibble( char n )
{
       databus &= 0xF0;
       databus |= n >> 4;
       _delay(5);
       setE;
       _delay(5);
       clrE;
}
```

The *lcd_send_nibble* function clears out the low nibble of PORTB while preserving the bit states of PORTB's high nibble. Other LCD functions that use *lcd_send_nibble* always send the byte to the LCD's most significant byte first. Thus, the nibble to send to the LCD (n) is always shifted to the lower nibble position of PORTB.

The HI-TECH C compiler includes the *_delay* function in a library that resides within the htc.h include file. The *_delay* function counts instruction cycles to produce a delay. With a clock of 20 MHz, that puts each instruction cycle at 200 nS. Thus, *_delay(5)* will produce a delay of 1,000 nS. The minimum E cycle time for the Newhaven NHD-0208BZ-FL-YBW LCD is listed as 500 nS with a minimum E pulse width of 230 nS. So, an argument of 5 in the *_delay* function is very conservative.

The *lcd_init* function is the first LCD function to call the *lcd_send_nibble* function:

```
void lcd_init(void)
{
     char j8;
     clrRS;
     clrE;
     clrRW;
        _delay(80000);
        lcd_send_nibble(0x30);
        _delay(30000);
        lcd_send_nibble(0x30);
        _delay(5000);
        lcd_send_nibble(0x30);
        _delay(10000);
        lcd_send_nibble(0x20);
        for(j8=0;j8<5;++j8)
{
             _delay(10000);
             lcd_send_byte(0,LCD_INIT_STRING[j8]);
        }
}
```

The sequence of nibbles, bytes, and time delays contained within the *lcd_init* function place the LCD in four-bit mode. Note the 0 in the argument of the *lcd_send_byte* function. An LCD command is sent with the LCD RS pin held logically low. A zero in this position of the *lcd_send_byte* function causes the LCD's RS pin to be driven logically low. Conversely, a one in the *lcd_send_byte* argument forces the LCD's RS pin logically high and the function's payload is designated as a character. Here's the source code for the second function to call the *lcd_send_nibble* function:

```
void lcd_send_byte( char address, char n )
{
       clrE;

          switch (address)
          {
                         case 0:
                                 clrRS;
                                 break;
                         case 1:
                                 setRS;
                                 break;
                         default:
                                 setRS;
                                 break;
          }
      _delay(5000);
       lcd_send_nibble (n);
       lcd_send_nibble (n << 4);
      _delay(5000);
}
```

The LCD in our design is configured physically as two lines with eight characters per line. Arranging the LCD characters as eight-byte arrays makes for an easy way to display LCD messages:

```
char lcdmsg_servo[]      = " SERVO   ";
```

With each message packaged as eight bytes, displaying SERVO on the upper line of the LCD is simple:

```
        char msg8;
        line1;
        for(msg8=0;msg8<8;++msg8)
              lcd_send_byte(1,lcdmsg_servo[msg8]);
```

Mixing in the Optical Encoder

Suppose we wanted to alternately display SERVO1 and SERVO2 by rotating the optical encoder's shaft. First, we would define our eight-byte messages. Our six-byte SERVOx

messages are centered between 0x20 space characters:

```
char lcdmsg_servo1[]
      = " SERVO1 ";
char lcdmsg_servo2[]
      = " SERVO2 ";
```

Next, we would clear the 16-bit position counter and limit our counts to 0 and 1. We must be sure to initialize both the high and low bytes of the POSCNT counter and the MAXCNT register:

```
        POSCNTH = 0;
        POSCNTL = 0;
        MAXCNTH = 0;
        MAXCNTL = 1;
        oldbyte = POSCNTL;
```

Our optical encoder code simply looks at the value of POSCNTL which will always be 0 or 1, and sends the message associated with the POSCNTL count value to the LCD:

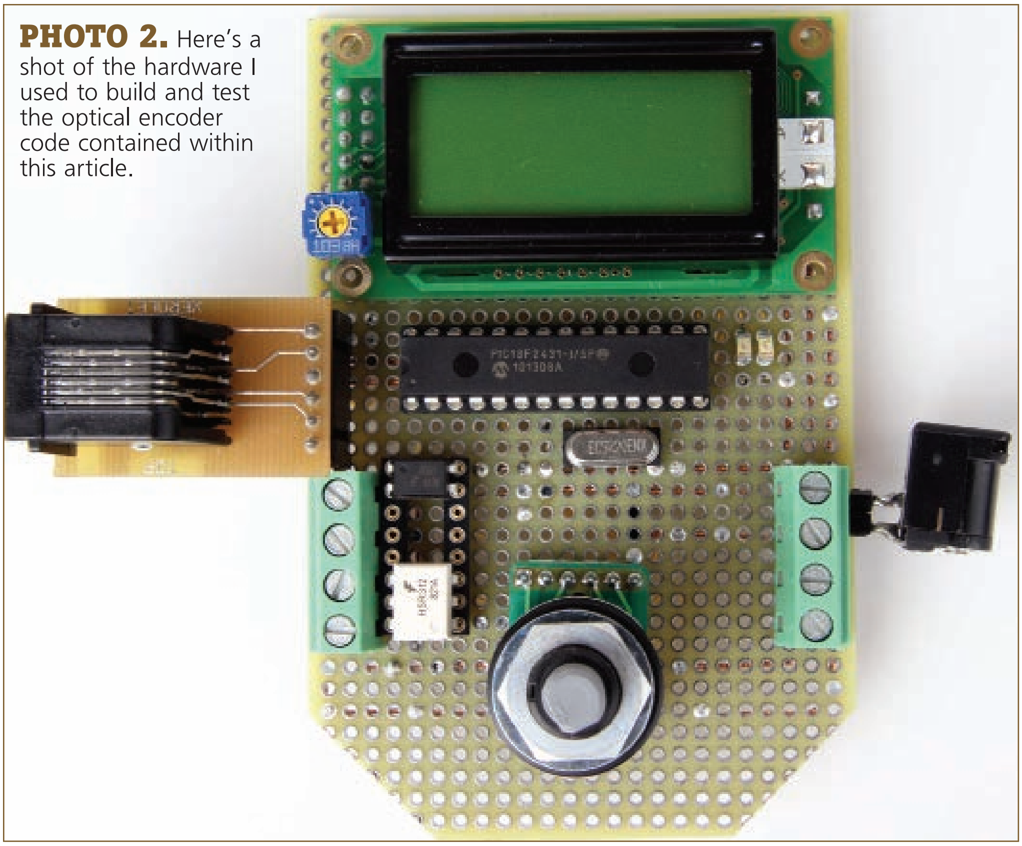


PHOTO 2. Here's a shot of the hardware I used to build and test the optical encoder code contained within this article.

```
do{
        if(oldbyte !=
POSCNTL)
        {
                switch(POSCNTL)
                {
                    case 0:
                        line1;
                        for(msg8=0;msg8<8;++msg8)
                             lcd_send_byte(1,lcdmsg_
                             servo1[msg8]);
                        oldbyte = POSCNTL;
                    break;
                    case 1:
                        line1;
                        for(msg8=0;msg8<8;++msg8)
                             lcd_send_byte(1,lcdmsg_
                             servo2[msg8]);
                        oldbyte = POSCNTL;
                    break;
                }
        }
}while(1);
```

As the optical encoder's shaft is turned, the value of POSCNTL will increment or decrement causing the oldbyte variable to differ from the new POSCNTL value.

To use the POSCNTL value to generate displayable numbers and text, we must convert the raw POSCNTL numeric value to an ASCII value. That is easily done by adding 0x30 to the POSCNTL value which is always less than 10 decimal. To keep the POSCNTL value below 10, we simply load MAXCNTL with 9 decimal:

```
        POSCNTH = 0;
        POSCNTL = 9;
        MAXCNTH = 0;
        MAXCNTL = 9;
        oldbyte = POSCNTL + 0x30;
        rotorbyte = oldbyte;
        lcd_gotoxy(1,1);
        lcd_send_byte(1,rotorbyte);
```

Notice we've added another variable called rotorbyte. The oldbyte variable will force a change just as it did in the SERVO alternate message code. The displayed character will be derived from the rotorbyte variable.

The *lcd_gotoxy* function will service one-line to four-line LCD modules with 20 characters or less per line. We are using it here to place the LCD cursor at the far left cell of line 1. Upon powerup or a CPU reset, a 9 will be displayed at this position. Here's the cursor positioning code:

```
void lcd_gotoxy( char x, char y)
{
// where x = lcd row (1,2,3,4) and y = column (1
thru 20)
    char address;
    switch (x)
    {
                case 1:
```

PIC18F2431 HI-TECH PICC-18 PRO	Microchip **www.microchip.com**

```
                address = 0;
                break;
        case 2:
                address = 0x40;
                break;
        case 3:
                address = 0x14;
                break;
        case 4:
                address = 0x54;
                break;
        default:
                address = 0;
    }
    address += (y-1);
    lcd_send_byte(0,0x80|address);
}
```

The LCD cursor placement math is based on LCD display address mapping. For a 2x8 LCD module, line 1 is addressed as 0 through 7 decimal and line 2 is addressed 40 through 47 decimal. Once the cursor is placed and the 9 is displayed at the cursor position, we can execute the 0-9 display code:

```
do{
     rotorbyte = POSCNTL + 0x30;
     if(rotorbyte != oldbyte)
     {
            lcd_gotoxy(1,1);
            lcd_send_byte(1,rotorbyte);
            oldbyte=rotorbyte;
     }
}while(1);
```

The rotorbyte will cycle from 0x39 to 0x30 with the first turn of the optical encoder shaft in a clockwise direction. The displayed count continues to increment as the shaft rotates in a clockwise direction. Rotating the shaft in a counter-clockwise direction will decrement the displayed count.

Rotating Your Encoder at Both Ends

You now have everything you need to add an optical encoder to the input end of your robotic projects. Process the optical encoder's pushbutton just as you would any other pushbutton attached to a PIC I/O port. The idea is to select your value with the rotation of the optical encoder shaft and lock in or save the value with a push of the knob that is set-screwed onto the optical encoder's shaft. I did just that with the optical encoder hardware you see in **Photo 2** and you can too. **SV**

Fred Eady may be reached via email at fred@edtp.com

Building a Real-Time GPS Tracking System

By Forrest Stanley

Adding a position reporting and tracking system to your robot comes with many challenges. It can be especially difficult providing live updated tracking information to a server or user. With a GPRS mobile connection, this problem can be overcome and even exploited to provide a mobile, live user interface to the tracking system. This article will detail how to build and add a tracking system to any mobile application, such as an autonomous robot or vehicle. This tracking system will utilize a GPRS/GPS module, a local web server, and a simple web interface created with the Google Maps API.

Adding a mobile GPRS connection with GPS functionality has the obvious navigational logging application. However, this functionality also enables options such as remote navigation, real-time web updates to a blog, or getting instant readings from the environmental sensors.

The mobile tracking system on Earth receives GPS data from the satellites to calculate its position.

Costs

With the info provided here, you will be able to build this device for about half of what many online stores charge.

For hardware, I used the NetBurner MOD5270LC Eclipse Ethernet Development Kit (**www.netburner.com** for $99). I also used the SparkFun Electronics GM862 Evaluation Kit (**www.sparkfun.com** for $230). For testing purposes, I would recommend purchasing the GM862 kit with an included power supply.

Hardware Used

This system uses the Telit GM862-GPS chip. This is a device that not only provides GPS signal serial output, but also handles the GPRS mobile connection. General Packet Radio Service (GPRS) allows the chip to use a mobile network service that can connect the device to a local web server. I used a NetBurner MOD5270 as a gateway between the GM862 and the end user. The MOD5270 initiates and manages the GPRS Internet connection, translates the GPS serial output into an object the web server can read, and

provides the TCP/IP stack. In addition to these two devices, you will need to run some web code on a local machine. I happened to use another NetBurner device, but any web server (including your PC) can work for this.

Using a Web Server

It is possible to use the MOD5270 built-in web server. However, I did not go this route as it requires an expensive static IP address for your mobile connection. A static IP address has a few downfalls, as well. If your device does not currently have a mobile connection, then the web page would not exist, and the end user would get a 404 "website not found" error. With a local web server, the application can always state the last known position of your robot.

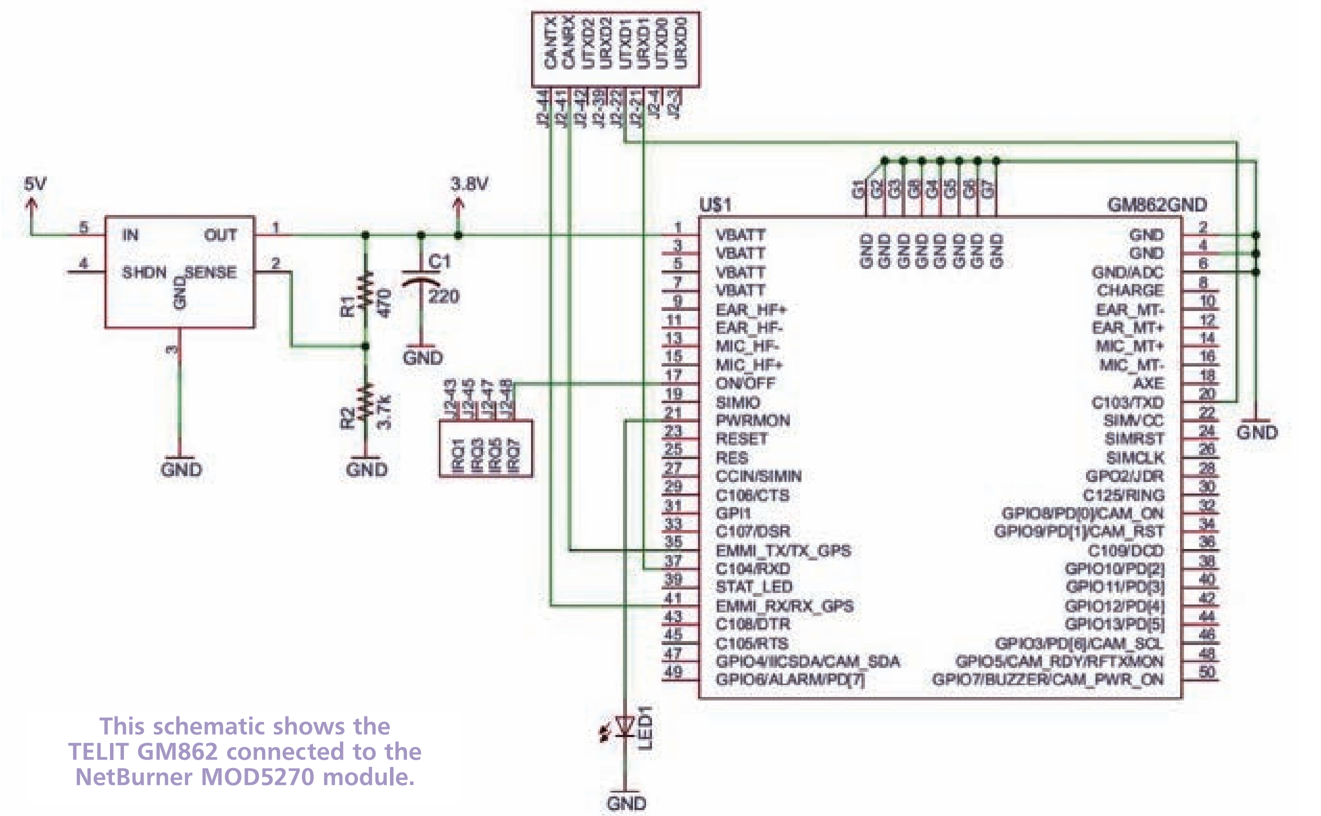
The web server interface uses the Google API to provide a familiar Google Maps' style interface. Google has provided a great API for creating simple GPS applications. With the Google API, the application's webpage is able to plot the robot's location anywhere in the world with both satellite imagery and easy to read maps. If your bot is near a photographed street location, then it also provides a panoramic eye-level view of the area.

Setting Up the Hardware

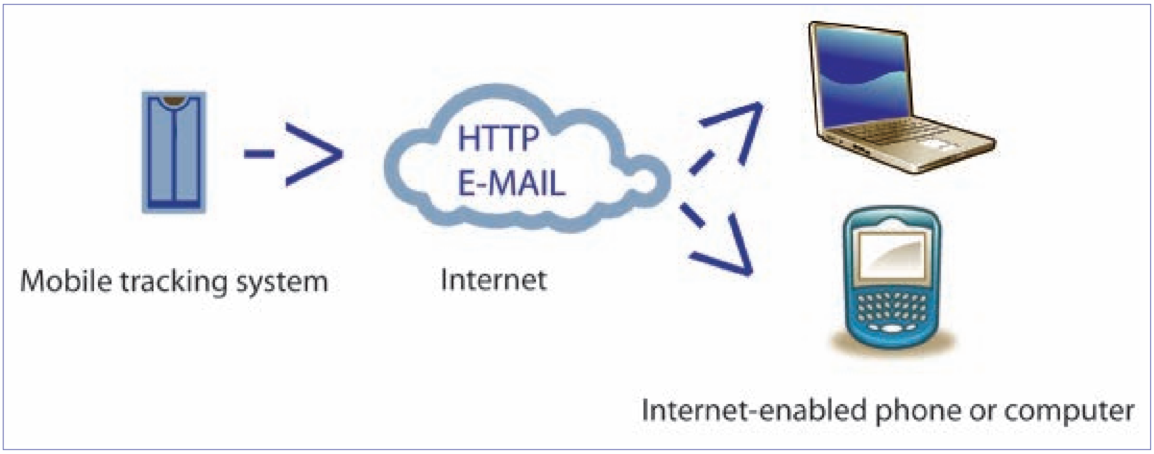
Hardware configuration is not too difficult. The biggest challenge is finding the 3.8V required by the Telit chip. In my setup, I had 5V and 3.3V power available. The **schematic** illustrates how I obtained 3.8V from the 5V source. The MOD5270 requires 3.3V. With 3.3V power already available, no voltage level translation is required.

The two devices require four lines for serial communication. Two of the lines will be reading the GPS values while the other line will be controlling the GPRS mobile connection.

Finally, you will need a way to toggle the Telit chip on and off. The on/off switch on the Telit device works exactly like a power button on a cell phone. It is a simple toggle that must be held for at least one second. Once held, the power will toggle either on or off. In the case of the Telit chip, you must pull the on/off switch down for at least one second to toggle the power.



This schematic shows the TELIT GM862 connected to the NetBurner MOD5270 module.



The server can send email updates or provide live updates on a webpage using Google Maps.

Establishing a Mobile Connection

The next step in this application is to establish a connection to the Internet via the GPRS device. The purpose of this web connection is to allow the current location of your robot to be sent to a web server that the user can view. The NetBurner tools provide some simple utility functions to open and manage the GPRS network connection. If any loss of signal occurs, the connection utility task will re-establish the network interface as soon as the signal returns.

A serial port connection is then opened from the NetBurner to the Telit chip. The Telit chip is always listening on the serial port, so begin by opening a serial port on the MOD5270. Once the MOD5270 and the Telit are communicating, you will need to send some basic initialization and configuration strings to the GPRS module. If you are familiar with AT commands, this is what the Telit chip uses for all configuration and dial commands. AT commands are configuration strings often associated with modems. Through AT commands, you will need to specify the mobile provider's dial-in number and login information. Along with the mobile provider's options, a country specific code is also required. Here's a configuration string that initializes the device for use in the US on T-Mobile's network.

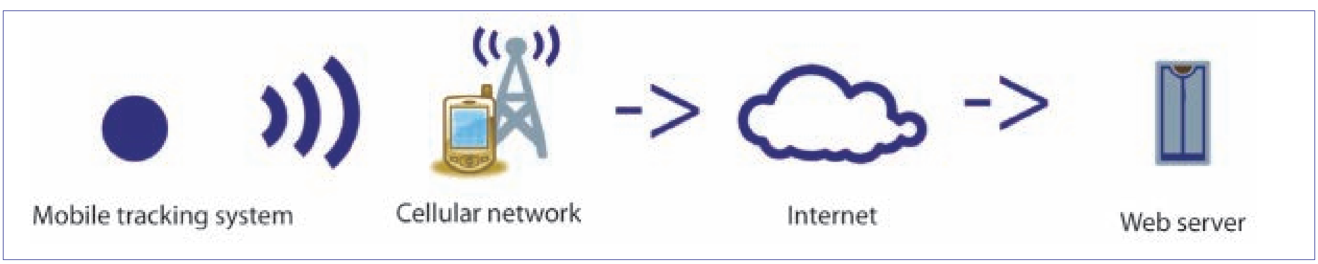
```
"AT+CGDCONT=1,\"IP\",\"internet2.voicestream.com\
"\r\n"
```

When "AT" is sent through serial, it indicates that a command string will follow. The CGDCONT command is used to set the GPRS network connection protocol. The command used indicates that the board will use an IP-based connection. Next, the APN (Access Point Name), "**internet2.voicestream.com**" is used. This address was provided by the mobile service that was used in this example.

Once the connection is configured, a dial AT command string is sent to the GPRS device.

```
"ATD*99***1#\r\n"
```

This dial string initiates the network connection that is always on. Unlike dial-up networking, no authentication is required in a GPRS connection. Your phone number is your identification, and is how the provider charges you for the service.



The mobile tracking system stores its position by sending data over a GPRS cellular network with Internet access on a publically accessible web server.

Once established, we allow the NetBurner GPRS network object to monitor the active connection. Mobile networking connections — like cell phone connections — are far from perfect. If the application loses signal, the network object will continuously search for a new signal, and re-establish the connection once one is found.

GPS Location Data

During the configuration and connection of the network, you should also be initializing the GPS module. Ideally, you want an update of the current location over serial every second.

Once turned on, the Telit GPS immediately begins its satellite search process to establish its current location. An antenna is a necessity in getting a good satellite read. Indoor use may degrade the signal, so in development, it's best to stay near a window with a clear view of the sky.

GPS readings come in the form of NMEA GPS sentences. NMEA is a protocol designed for communication between maritime navigational instruments. GPS communication is defined within this protocol.

A pitfall at this point would be failing to configure the GPS module to display output. By default, the device will not output any location information. For my application, I am interested in getting the signal quality, current time, latitude, and longitude. You are not limited to this output though. You can configure the device to display speed, direction, altitude, and other navigational readings. The readings I require can be derived from the NMEA GPGGA sentence, so we will enable it with a special Telit configuration command.

```
int i = siprintf(buff, "$PSRF103,00,00,01,01*");
       // Enable GPGGA
siprintf(buff + i, "%02X\r\n\0",
       GetGPSCheckSum(buff));
writestring(fdIncoming, buff);
```

This command enables the GPGGA sentence and instructs the GPS module to send this location statement every second. The GPGGA sentence arrives from the Telit GPS in the following example format.

```
    $GPGGA,063214.000,2125.1295,N,00903.9206,E,1,
03,2.1,-108.0,M,46.1,M,,0000*4E
```

Your application must parse this data string. The only information that I need to save from this statement is the current time, latitude, longitude, and satellite connection quality. Everything else can be dropped. The connection

GPS

The Global Positioning System (GPS) is a worldwide navigational tool developed and maintained by the US Department of Defense. It uses orbiting satellites that transmit specially coded signals which are picked up by receiver units. With the information broadcast from the satellites, these receiver units can compute position, velocity, and time.

The GPS system consists of three main segments. The space segment details the satellite system. These space vehicles are in constant orbit around the Earth. Five to eight of these satellites are visible from any point on the Earth's surface. The control segment of GPS consists of tracking the flight path and updating the coded signals of the satellites in orbit. This segment is maintained by the US Air Force. The final — and most important for us civilians — is the user segment. The user segment consists of all integrated and dedicated GPS receiver units and the community of users around these devices.

The GPS receiver figures out its position by triangulating the distance between satellites that it picks up. At least four satellites must be in view to figure out the four dimensions X, Y, Z (Position), and Time. Increasing the number of satellites available to the unit will increase the accuracy of your fourth dimensional location.

GPRS

General Packet Radio Server (GPRS) is a wireless communications standard that uses a mobile cell phone connection primarily to provide a network address to your robot. With a GPRS-enabled device, it is possible to connect to the Internet, view video, read and write emails, and send and receive packets of data.

GPRS is available on Global System for Mobile Communications (GSM) which is part of the Second Generation (2G) wireless telephone technology. While it does not provide the speed of a Third Generation (3G) device, it will provide a much wider coverage area. The connection speed of a GPRS will increase and decrease with current wireless signal quality. With good coverage, a GPRS device can attain speeds up to 112 kbps.

The network connection for the mobile networking device is handled by an access point. This access point has several channels available to offer the mobile device. In a standard GSM connection, the device and the access point utilize a circuit-switched connection. A circuit-switched connection dedicates one channel to the connection of the device and remains in an always-on state until the connection is terminated. A GPRS capable device will utilize a packet-switched connection to the access point. In this connection type, multiple channels are opened between the device and the access point, allowing much higher throughput. Packets are sent out on an as-needed basis, allowing the access point to host multiple devices on the same channel.

quality is saved so that if the application cannot see any satellites, it stops updating the GPS location of the robot. If this is not done, you will start reading invalid values.

Building the Web Page

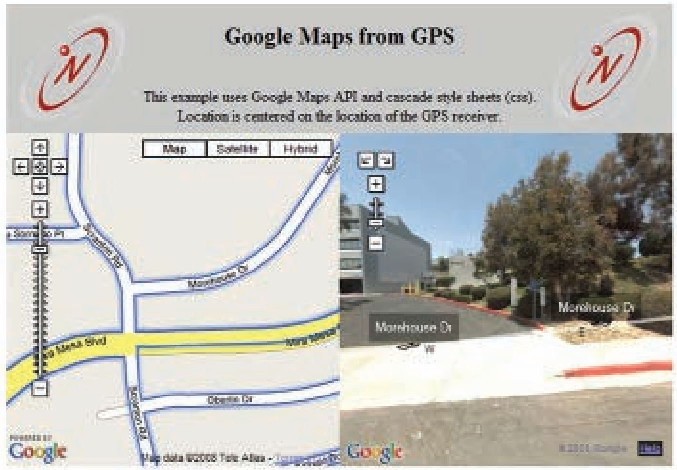
Once you know the location of your bot, it is time to map out the location in the Google Maps API. The Google Maps API lets you embed Google Maps in your own web page. With a latitude/longitude, the server is able to map the location of the device on a web page. If the device is located on a street that supports Google Street View, then this is also drawn on the web page. (See the **figure** for the web page example.)

The web page that is displayed to the user is separated into two different maps. On the left, a map is centered on the area that was last received by the local GPS location object. On the right, a panoramic picture of the nearest street location supported by Google Street View is displayed. If no nearby street is supported by Google Street View, the image is not rendered.

Initializing, drawing, and centering the map are completed with two lines of code calling the Google API.

```
        var map = new GMap2(document.getElement
        ById("map_canvas"));
        map.setCenter(new GLatLng(<!
        —FUNCTIONCALL getLat —>,<!—
FUNCTIONCALL getLon —>), 16);
```

The FUNCTIONCALL comment in the HTML code is a utility function provided by NetBurner that allows HTML code to call C/C++ functions prior to rendering the web page. In this case, the functions getLat and getLon are called. These are functions that return the most recent



latitude and longitude in plain text.

Creating the panoramic view works in a nearly identical way. The difference is that instead of creating a GMap2 object, a GStreetviewPanorama object is created. Once created, you center the view on your location with a setLocationAndPOV API call.

Additional Capabilities

Many options exist to extend this application with a little work. Logging incoming data to your web server will allow you to create a path in Google Maps to track the entire journey of your robots. Directions could be sent to the robot via text messages, turning any phone into a navigational device. **SV**

Parts

MOD5270LC Development Kit
Netburner NNDK-MOD5270LC-KIT

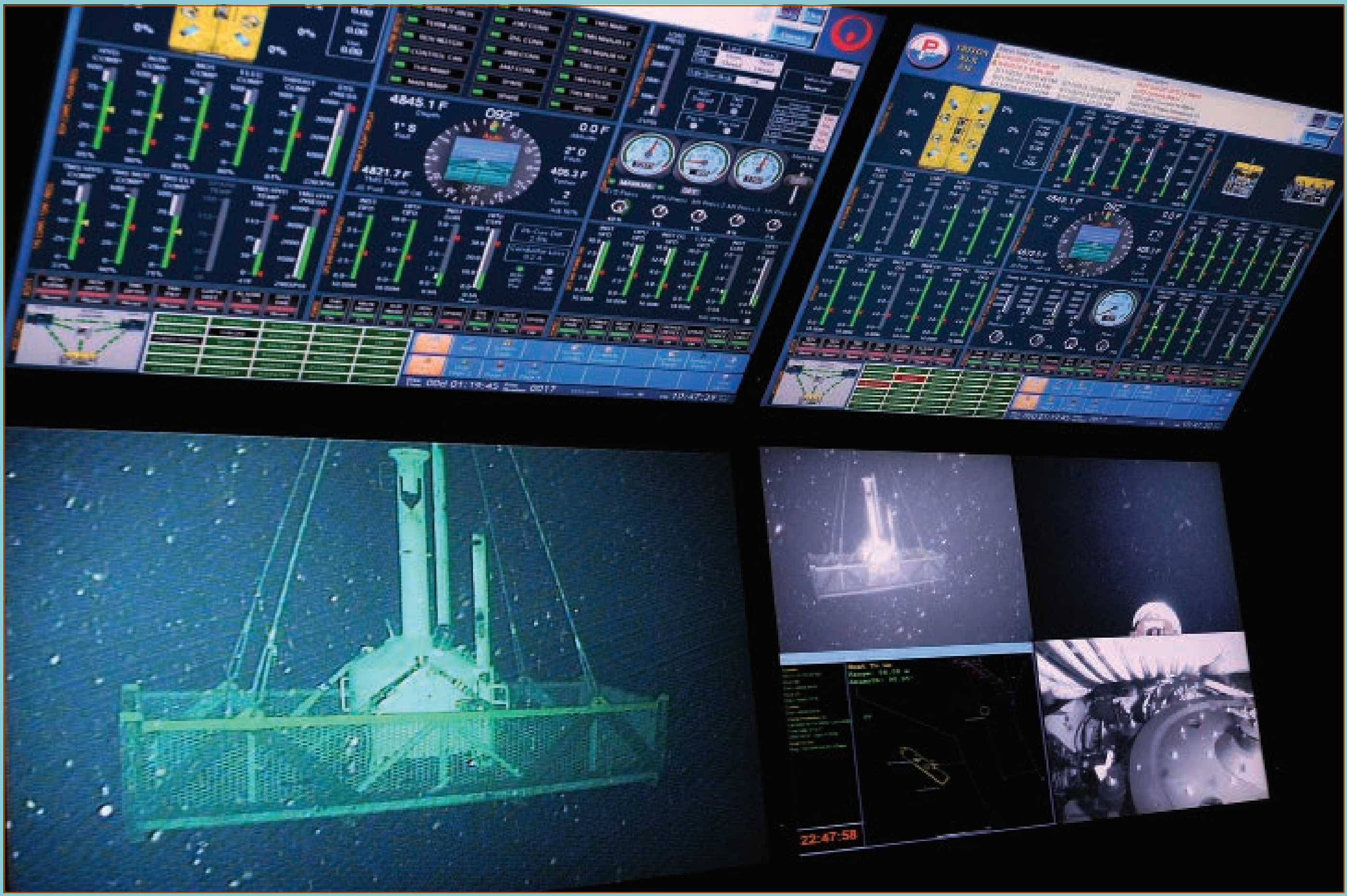
GM862 Evaluation Kit; SparkFun CEL-00479

Robot Rescue:

ROVs Lend a Hand During Gulf Oil Spill Disaster

ROVing Around Davy Jone's Locker Isn't Tough to Fathom

By Dave Prochnow



Video acquired by a remotely operated vehicle is monitored and recorded onboard the motor vessel Viking Poseidon. The video display shows the small pollution containment chamber known as the Tophat. *[U.S. Coast Guard photo by Petty Officer 3rd Class Patrick Kelley.]*

Imagine trying to flick a switch with a set of claw-like fingers from one mile away via an umbilical tether that is connected to a robotic arm. Now, imagine performing this delicate operation under the groaning weight of 2,400 PSI — that's over one ton of pressure per square inch. Throw in a couple of other unimaginable factors like operating in a claustrophobic inky blackness with temperatures hovering around 34°F, and contending with a sea current swaying your arm back and forth.

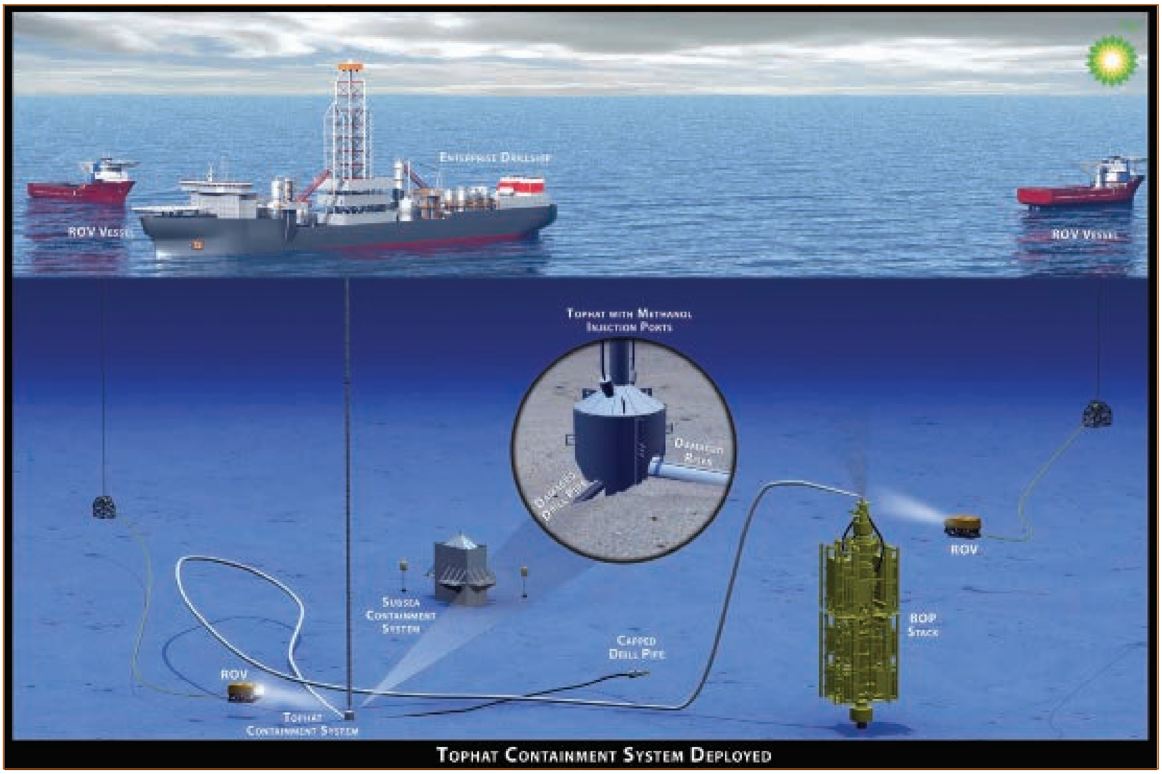


Illustration of Tophat Containment System deployment.

Graphics courtesy © BP p.l.c unless noted otherwise.

Sounds Pretty Arduous, Doesn't It?

Oh, and there's one more thing: That switch you need to flick is connected to a sunken oil rig's blowout preventer. A blowout preventer (BOP) that is guarding a wellhead which is leaking roughly 5,000 barrels (210,000 gallons/795,000 liters) of crude oil per day. Yep. Just another day in the life of the undersea remotely-operated vehicle.

The only difference with this particular scenario is that this "day in the life" has stretched into weeks as the tragic explosion and sinking of the Transocean Deepwater Horizon oil rig [officially known as British Petroleum (BP) oil well Mississippi Canyon 252 or MC252] approximately 40 miles off the southern coast of Louisiana between April 20-22, 2010 resulted in an uncapped, leaking wellhead.

NOTE: This article is not a critique of the Deepwater Horizon oil spill disaster response nor is it an indictment of the offshore oil drilling industry's environmental impact.

Send in the Robots

Within hours of this tragedy, BP had launched a small fleet of remotely-operated vehicles (or ROVs) in an underwater rescue attempt at triggering the recalcitrant BOP. Using pliers, saws, and claws, the ROVs vainly tried to flick that magical switch on the blowout preventer that would cap the wellhead.

After several days of poking and prodding, the ROVs were able to effectively shut off one of the three leaks emanating from the destroyed oil rig. Unfortunately, this leak was associated with a hole inside the crumpled main riser pipe which transports crude oil from the wellhead to the oil rig on the ocean's surface and not the BOP.

Following on this minor success, BP increased its ROV armada to 12 robots and began working on three different attempts at stemming the flow of oil from the wellhead.

The first attempt was trying to flick that switch — the failsafe safety switch which would activate the BOP and immediately seal the wellhead. This attempt was ongoing during the entire response effort.

The second attempt performed by the ROVs seemed almost counterintuitive. While continuing the BOP salvage efforts, a second squad of ROVs were preparing the BOP to be killed. Known as a "top kill," this response would effectively cram the unresponsive BOP with junk — including golf balls and automobile tires called "junk shot" — then "seal" the deal with a stream of specialized heavy fluids called drilling mud that would prevent continued oil flow through the BOP. In an oversimplified analogy, a top kill would perform a result similar to that which would be obtained from a functional BOP.

While two groups of ROVs were occupied with removing control panels, mending some cabling, and fiddling with the failsafe switch, a final group of robots orchestrated an entirely different attempt at curbing this looming ecological disaster.

A small dome called "Tophat" was shipped out to the accident site and lowered to the seabed in preparation for deployment. Similar to its much larger cousin called a containment dome which had frozen up and failed on May 8, 2010, Tophat was designed to minimize the deleterious effects from freezing hydrates. While this system had never been used in water depths of 5,000+ feet, BP unleashed a small cadre of ROVs for shepherding Tophat into place while warning that its successful operation was "uncertain."

Main oil leak at end of riser pipe/12 Inch wrench and ROV in background.

Whose Robot's Whose?

Oddly enough, not all of these ROVs swimming around the wellhead are owned or operated by BP. Huh? Yup. Unlike other industries that competitively feed off of each other's misfortune, in the oil industry everybody lends a hand in taming a disaster. So, when an ROV is under contract with a competitor and a tragic event mandates swift undersea help, a staunch corporate foe will quickly become your strongest ally.

In the case of the MC252 accident, ExxonMobile became a strong supporter in providing ROV support. Likewise, Chevron lent a hand in dealing with the cantankerous BOP, while Shell sent a small flotilla of surface support vessels.

BP ROVs Role In a Nutshell

ROVs have been used in deepwater industries for more than 30 years, mostly to carry out routine maintenance and construction work.

ROVs are linked to the surface by means of a tether (or "umbilical") which is essentially a group of cables that are used to carry electrical power, video, and other data signals between the ROV and the operator on a vessel or onshore.

An operator can control the movements of these highly maneuverable machines under the sea, performing a variety of difficult operations from the safety of a vessel or dry land. A key benefit of an ROV is that it can carry out work without the need for human divers. Furthermore, ROVs can work beyond the depth and pressure that a human could safely dive.

An ROV's arms or "manipulator," are easily controlled and have been developed over the years to have the same range of motion as a human arm. In fact, some are so sophisticated that they could, theoretically, pick up an egg without breaking it. To direct the arms, the ROV operator at the surface holds a joystick. Some ROVs even have force feedback mechanisms so that the operator can feel resistance and reaction to the force being applied.

ROVs are frequently equipped with a number of systems to aid in navigation. Most have lights and video cameras. Some ROVs, in addition, have sonar systems which help operators navigate safely through undersea obstacles and terrain in murky waters. GPS systems are also available to help track an ROV's exact location.

The vehicles typically carry a variety of specialized tools that are designed for compatibility with subsea equipment. Using these tools, ROV operators can manipulate the valves and controls of sub-sea well heads.

At least 12 ROVs were deployed for various purposes in addressing the MC252 spill in the Gulf. Some are engaged in monitoring the well. Others are supporting attempts to activate the blowout preventer. In other tasks, for example, an ROV was being used to hold a wand to spray dispersant into oil on the seafloor near the leak in the main riser pipe.

A robotic arm of a Remotely Operated Vehicle (ROV) attempts to activate the Deepwater Horizon Blowout Preventer (BOP), Thursday, April 22, 2010. In addition to the use of ROVs, the unified command is mobilizing the Development Driller III, a drilling rig that is expected to prepare for relief well drilling operations, to stop the flow of oil that has been estimated at leaking up to 1,000 barrels/42,000 gallons a day.
[Photo courtesy U.S. Coast Guard Eighth District External Affairs.]

Regardless of who really "owns" any particular ROV, all of the undersea work was choreographed through a Deepwater Horizon Response Command Center. From this command center, tasks were doled out to several surface motor vessels who actually "handled" the operation of each ROV. For example, at one time during the early days of the BP oil spill response, six ROVs were hovering around the BOP, each tethered to their respective surface support ship. These ROVs were equipped with video cameras and an array of "hand" (more like claw) tools like wire cutters, pliers, and "hot stabs." A hot stab is a metal connector that can plug directly into a hydraulic system for subsequent control from the topside support ship.

In the fleet of over 700 ships that responded to the MC252 leak, it's tough to single out one support ship over another. But the Viking Poseidon (Ulstein SX121) could be considered an exception to that statement. And there is no mistaking her in the Gulf — she's the big orange giant.

Viking Poseidon is a highly specialized deepwater marine vessel owned and operated by Veolia ES Industrial Services. The Poseidon's design technology and equipment made it particularly well-suited for handling this deepwater crisis. Ironically, the Viking Poseidon had just been delivered to Galveston Bay, TX in February 2010 and had completed its first deepwater job in the Gulf of Mexico on March 26, 2010.

Mr. Michel Gourvennec, President and Chief Executive Officer of Veolia Environmental Services North America, stated, "The Poseidon clearly complements the capabilities of our expanding Marine Services division. As the market for deepwater capabilities — particularly in the Gulf of Mexico — continues to grow, the Poseidon becomes a strategic asset that will put Veolia at the forefront of the marine services industry. We are happy to add her to our fleet."

The Poseidon is the largest Ulstein X-Bow vessel in the world. It was built by Ulstein Verft for Eidesvik Offshore in Norway during 2008. It is a 130 meter long, 25 meter wide vessel with a deck area of 1,720 square meters. The Poseidon has a maximum speed of 14 knots and can accommodate a crew of up to 106. Its advanced features — which allow it to deploy heavier loads and operate further offshore — include two work class ROVs with launch and recovery systems (LARS), an HMC-250T active heave compensated knuckle boom crane, and a 15 ton electro-hydraulic deck crane with folding boom.

Luckily, the Viking Poseidon was immediately available to the Deepwater Horizon Response Command Center. In fact, it was the Viking Poseidon that was tasked with lowering Tophat into place next to MC252's broken main riser pipe on May 11, 2010.

ROV for Hire

Other than guiding Tophat into place, other ROVs were being used for several, much more specialized tasks:

Spraying. One of the chief deterrents against oil spills has traditionally been the application of a dispersant. This chemical — when sprayed on oil — causes it to be disassociated into smaller droplets. Unlike a surface oil spill, however, the leak from MC252 was so deep in the ocean that an ROV had to be deployed directly next to each oil

Viking Poseidon left Ulsteinvik to be fitted out with a 250 tonne offshore crane just before Christmas 2009. On January 13, 2010 the vessel was delivered to Eidesvik Offshore. *[Photograph courtesy of Ulstein Group.]*

It's an ROV World, After All

You can follow the ROV industry including careers, equipment, and news at ROVWorld:

www.rovworld.com/index.php

leak site with a specialized spraying wand clutched in an articulated arm. This enabled the dispersant to be injected directly into the oil leak. Over 590,000 gallons of dispersant had been used during the first half of May.

Inspection. Following the sinking of the Transocean Deepwater Horizon rig, the ocean floor was littered with over 5,000 feet of twisted, mangled pipe that constituted the rig's main riser pipe. It was through this pipe that several leaks were gushing crude oil into the surrounding water. ROVs equipped with video cameras were employed to scour every square inch of this pipe, relaying this vital visual data via fiber optic cable to engineers at the Response Command Center.

Manipulation. Since the first day following the accident, engineers were convinced that the fastest and easiest way of stopping the oil leak was engaging the ram locks on the BOP. ROVs relentlessly toiled at activating the BOP. This effort was ongoing, even while other attempts were being made at stemming the flow of oil from MC252.

Construction. As incredible as it may sound, while all of these other salvage tasks were being performed, another set of ROVs were being used for building a shutoff valve inside the leaking main riser pipe. Actually, this work is more along the lines of a traditional ROV role — tirelessly toiling underwater building a widget faster than it would take human divers to complete. And at nearly one mile down, this time savings is extremely valuable. According to the ROV industry, ROVs can work for days, whereas divers can only endure several hours underwater. Likewise, a skilled ROV operator can perform almost every function that can be performed by an experienced diver.

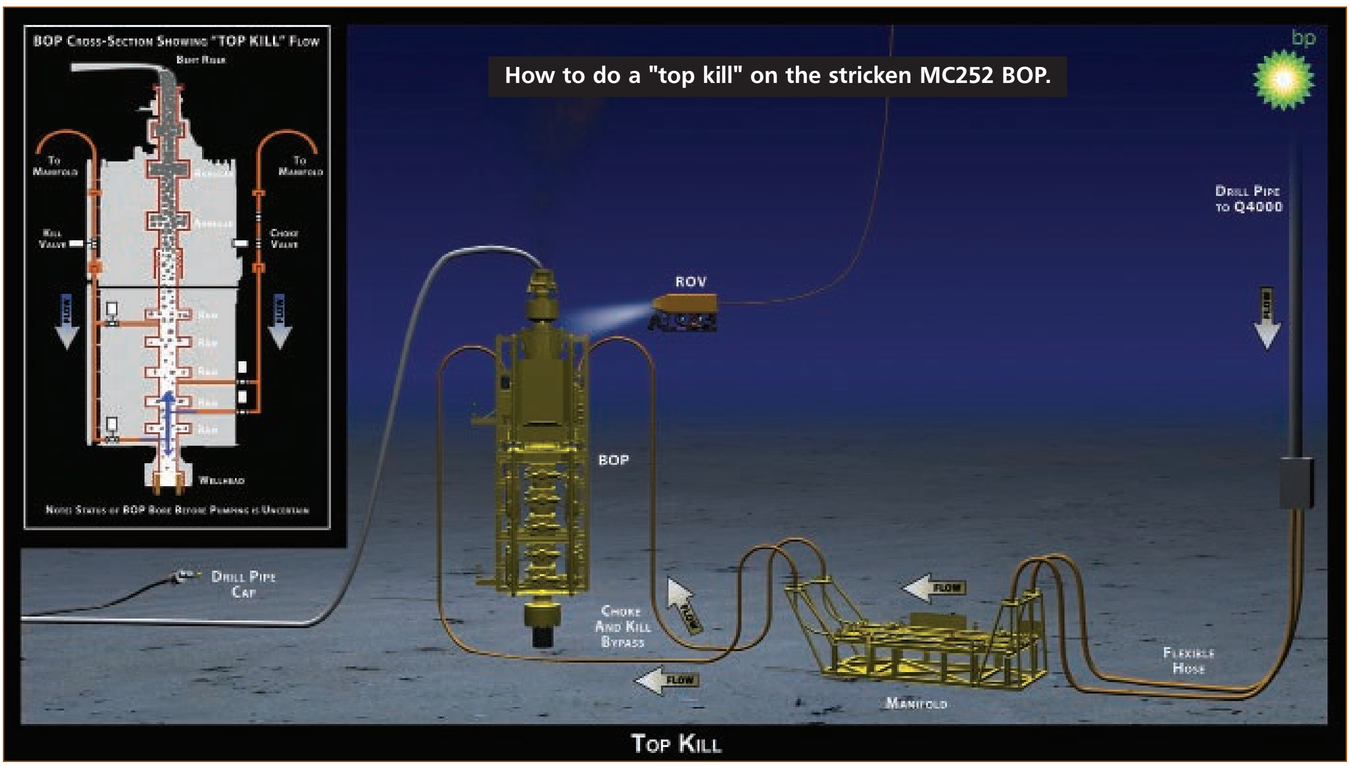
This entire operation doesn't come cheap. During the height of the Deepwater Horizon response effort, BP was spending over $6 million per day.

Stanching the Flow

By May 17, 2010, the round-the-clock work of BP's engineers was beginning to pay off. A five foot long section of pipe had been equipped with rubber seals for holding back seawater. This rubber-fitted capture pipe — known as

Terminology Used in Oil Spill Containment

BOP — Blowout preventer.
Hydrates — Ice-like formations of methane and water that occur in seawater at low temperature and high pressure.
Junk shot — Clog the BOP with golf balls and rubber tire pieces prior to attempt a top kill.
RITT — Riser insertion tube tool.
ROV — Remotely-operated vehicle.
Tophat — A containment funnel used to channel oil to a surface collection/processing vessel where the oil is separated from seawater.
Top Kill — Filling the BOP with heavy drilling mud followed by capping the well head with cement.



How to do a "top kill" on the stricken MC252 BOP.

DIY ROV

If you'd like to get your hands dirty, err, wet with some hands-on experience piloting your own ROV, there is a set of detailed construction documents available from Robert Gordon University (RGU) School of Engineering. Dubbed the RGU ROV, this experimental model is claimed to have been derived from BEAM-like technology and is designed for shallow freshwater environs like a bathtub or swimming pool.

www4.rgu.ac.uk/eng/robotics/page.cfm?pge=8192

a riser insertion tube tool (RITT) — was then connected to a one mile long section of pipe leading up to the drill ship, Discoverer Enterprise (which can hold approximately 5 million gallons of oil). Using ROVs, the RITT was inserted into the damaged 21 inch diameter well head riser pipe that was leaking the oil.

According to Kent Wells, senior executive vice president of BP, " ... we do have oil and gas coming to the ship now, we do have a flare burning off the gas, and we have the oil that's coming to the ship going to our surge tank."

This small success was not achieved without its share of failure, however. BP had initially begun trying to insert the RITT on May 14, but the connection to the drill ship failed and the entire tube/pipe assembly had to be brought back to the surface for adjustments.

Between May 15 and May 16, the RITT was once again inserted into the damaged riser pipe and was working for approximately four hours when it was dislodged by a rogue ROV that was "mishandled" by its operator. Even though that second attempt failed, it proved that BP engineers had concocted a capture design that could keep dangerous freezing hydrates from forming, Mr. Wells said.

Following the third — and successful — RITT insertion attempt on May 17, the Discoverer Enterprise began receiving just over 1,000 barrels of crude oil (approximately 42,000 gallons) per day. After 24 hours of operation, the RITT crude oil intake amount had been increased to about 2,000 barrels per day or roughly 20% of the total leak volume; within a week that volume had swollen to about 3,000 barrels or 126,000 gallons of crude oil per day. Meanwhile, BP continued its work on drilling two relief wells (one began May 2 and the other was started May 16), hoping to completely stop the oil leak from MC252. Each of these wells would take three months to finish.

At press time for this article, neither of these wells had been completed. **SV**

Meet the ROVs

No matter which task is at hand or claw, one of the workhorse ROVs used throughout the response effort was the Triton. The Triton line of ROVs are designed and manufactured by Perry Slingsby Systems (PSS) which is a subsidiary of subsea technology and services giant, Triton Group. Featuring a strong lineup of heavy duty work ROVs, PSS unveiled a new lightweight Triton system — the Triton XLR — in late 2009 that was a curious departure from earlier models.

This departure was explained in a statement released by Triton Group CEO, Martin Anderson: "The Triton XLR is the next step in the evolution of subsea systems which started with the XL. Our aim was to create an innovative system which offers the utmost in efficiency and reliability for the lightweight end of the market.

"The industries in which we operate are in a state of continual change and the requirement for new design and engineering innovations is constant. Market driven research and development underlies all of our current and potential future products, ensuring our expertise is applied to the areas today which will take our business — and our customer's business — to where it needs to be tomorrow.

"PSS has led the industry in all respects and continues to do so today with advanced, robust, and dependable ROVs in the world."

Following that 2009 release, there are now three ROV models in the PSS Triton lineup:

Triton® XLR 100/125

A medium-duty work-class ROV. Described as the "baby brother" to the heavy-duty Triton XLX, this 125 hp ROV was designed for undersea drill and light construction tasks with a state-of-the-art integrated real-time control engine (ICE) featuring a GUI and ergonomic pilot (operator) control console with touch screen operation.

NOTE: At press time, no Triton XLR ROVs had been used in the Deepwater Horizon response.

Vehicle Specifications
Dimensions: 2,780 mm x 1,500 mm x 1,800 mm
Weight in air: 2,850/3,450 kg
Depth rating: 2,000/3,000/4,000 meters
Payload: 150 kg
Horsepower: 100/125 hp
Tooling Packages
Three cameras
Tritech intelligent gyro compass
Digiquartz depth sensor
6 x 250 watt lights
Electric pan/tilt unit
12 Station main manifold
Two Off thruster manifolds
Piloted high-flow tooling manifold
Seven-Function manipulator
Five-Function grabber

Triton® XLS

A heavy-duty work-class ROV. Featuring heavy-lift 750 meter tether management system (TMS) and fitted with a Schilling seven-function Titan 4 (T4) and five-function rigmaster manipulators, the XLS is supported with a specialized Dynacon 6022 launch and recovery system.

Vehicle Specifications
Dimensions: 3,000 mm x 1,850 mm x 1,890 mm
Weight in air: 4,400 kg
Depth rating: 3,000/4,000 meters
Payload: 300 kg
Horsepower: 150 hp
Tooling Packages
Gyro/Fluxgate compass
Pitch/Roll sensor
Digiquartz depth
Doppler velocity
General function valve pack
Lighting system
Left/Right manipulator
Sonar
Hydraulic tool interfaces

The Triton XLS 12.
[Photo courtesy of Triton Group.]

The Triton XLX 01. *[Photo courtesy of Triton Group.]*

Triton® XLX

A heavy-duty work-class ROV. This deep water 250 hp workhorse includes ICE, GUI, redundant computer systems, gigabit Ethernet telemetry, advanced, interactive diagnostics, and a 4 x 40" HD Plasma video wall control system.

Vehicle Specifications
Dimensions: 3,226 mm x 1,803 mm x 2,000 mm
Weight in air: 4,900/5,600 kg
Depth rating: 3,000/4,000 meters
Payload: 250-550 kg
Horsepower: 150-250 hp
Tooling Packages
Gyro/Fluxgate compass
Pitch/Roll sensor
Digiquartz depth
Doppler velocity
16-Station proportional main manifold
14-Station proportional thrusters manifold
12-Station proportional tooling manifold
2 x 250-watt lights

ROVs aren't strictly a PSS mainstay. There are other ROV manufacturers that lend a hand in undersea drilling operations:

Oceaneering International (**www.oceaneering .com/rovs/millennium-plus-rov/**)

Millennium® Plus

A dual manipulator, cage deployed ROV with enhanced thruster configuration, more serviceability, tooling flexibility, simple survey integration capabilities, microprocessor-based telemetry, and fiber optic transmission link for all video and data signals between the vehicle and the surface control vessel.

Vehicle Specifications
Dimensions: 11'5.5” x 5'4.5” x 6'
Weight in air: 8,800 lbs
Depth rating: 10,000 ft (standard)
Payload: 900 lbs
Horsepower: Two ea 110 hp hydraulic power units (220 hp)
Tooling Packages
1.25 in wire rope cutter
1 in fiber rope cutter
Ring gasket replacement tool package
TP03 dredge/jet pump
Rotary grinder/cutter/buffer
1 x 2,500 psi @ 3.5 gpm intervention pump
2.5 gal intervention pump reservoir
10,000 PSI high pressure intervention package

Phoenix International (**www.phnx-international.us/**)

Remora 6000

Not every ROV is a wrench-wrangling workhorse. This is a deepwater, lightweight light-duty ROV that is optimized for drilling and construction observation.

Vehicle Specifications
Dimensions: 1.7 m x 1 m x 1.2 m
Weight in air: 900 kg
Depth rating: 6,000 meters
Horsepower: 25 hp
Observation Packages
2 x Hydro-Lex six-function manipulators
Prizm video
Simrad 1367 color CCD video camera
Simrad 1324 ultra-low light SIT camera
Simrad scanning sonar

Phoenix International Remora ROV on deck. *[Photo courtesy of Phoenix International.]*

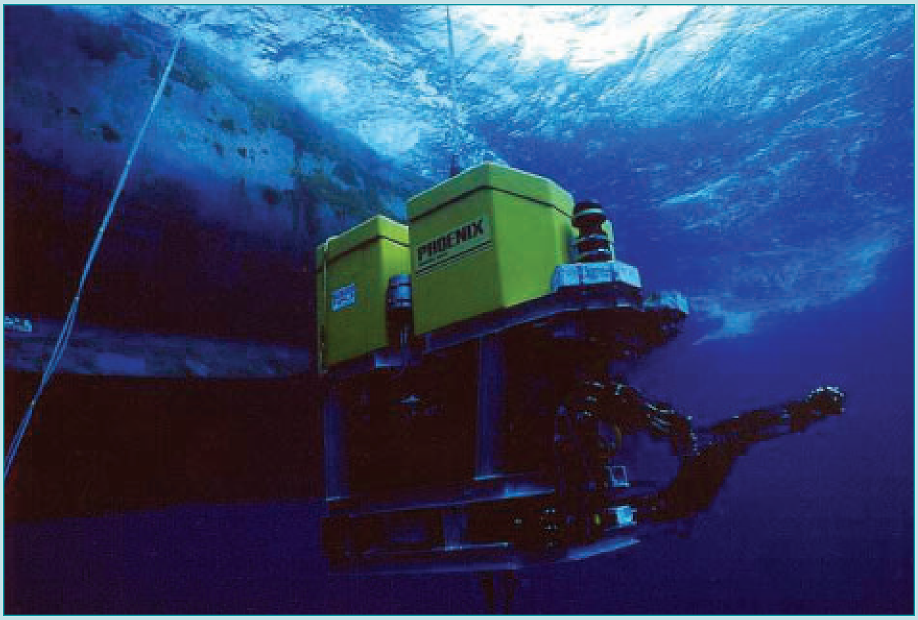
Laser gyro
Simrad altimeter
Depth sensor

Sonsub (**www.sonsub.com/rov/**)

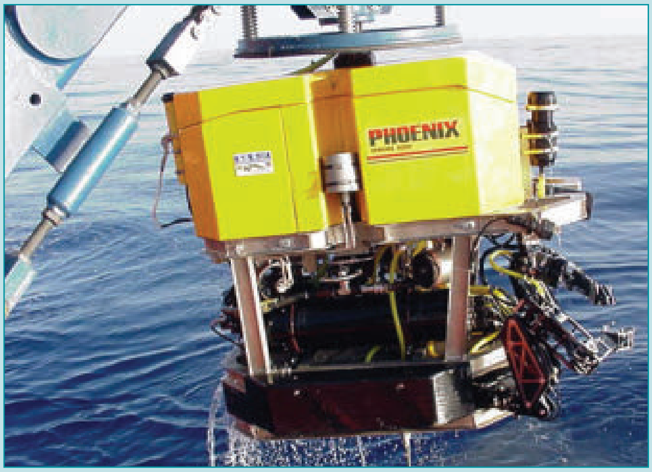
Innovator Leviathan

Powerful 250 hp heavy work-class ROV featuring a heavy weather launch and recovery system. Featuring a dual-shaft electric motor powering four 17" horizontal thrusters and three 15" vertical thrusters, Leviathan is specifically designed for sustained power and control for deepwater operations. A stable work platform, the Innovator Leviathan is engineered to operate continuously in harsh weather conditions, high current, or extended dive conditions.

Vehicle Specifications
Dimensions: 136" x 64" x 84"
Weight in air: 10,000 lbs
Depth rating: 3,500 meters
Horsepower: 250 hp electric motor
Tooling Packages
Seven-function manipulator
Five-function grabber
Fiber optic gyro
Surface processing sonar
Digital sonar head
Eight cable/connectors for fiber optic video camera



Phoenix International Remora ROV. *[Photo courtesy of Phoenix International.]*



Launch and recovery of the Phoenix International Remora 6000. *[Photo courtesy of Phoenix International.]*

Sonsub's Innovator 250 ROV. *[Photo courtesy of Sonsub.]*

Veolia Industrial Services (**http://veoliaes-is.com/Services/Special-Services/Remotely-Operated-Vehicle**)

Veolia Environmental Services Marine Services (VES) — a special service of Veolia Industrial Services — employs both inspection and work-class ROVs in its offshore and inland marine construction, repair, and inspection work contract services. Noteworthy in their ROV inventory are several PSS Tritons. Examples of these Tritons include:

Triton XLS-26 Work-Class ROV
Triton XLS-17 Work-Class ROV

VES has recently acquired two Triton XLX Heavy Duty Work-Class ROVs. This acquisition enables VES to support drilling operations to depths of 13,120 feet (4,000 meters).

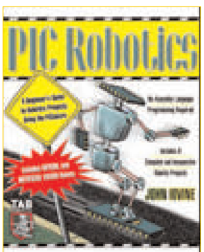
The *SERVO* Webstore

Attention Subscribers ask about your discount on prices marked with an *

CD-ROM SPECIALS

ROBOTICS

PIC Robotics
by John Iovine

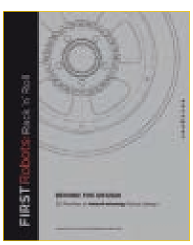


Here's everything the robotics hobbyist needs to harness the power of the PICMicro MCU!

In this heavily-illustrated resource, author John Iovine provides plans and complete parts lists for 11 easy-to-build robots each with a PICMicro "brain." The expertly written coverage of the PIC Basic Computer makes programming a snap – and lots of fun.
$24.95

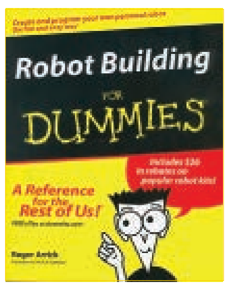
FIRST Robots: Rack 'N' Roll: Behind the Design
by Vince Wilczynski, Stephanie Slezycki

More than 750 photographs!



The second annual book highlighting the creativity and process behind 30 winning robot designs from the 18th annual international FIRST Robotics Competition. The FIRST organization, founded by Dean Kamen (inventor of the Segway), promotes education in the sciences, technology, and engineering.
Reg $39.95

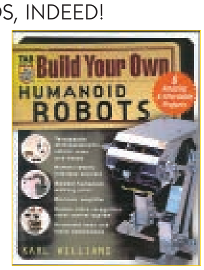
Robot Building for Dummies
by Roger Arrick / Nancy Stevenson



Discover what robots can do and how they work. Find out how to build your own robot and program it to perform tasks. Ready to enter the robot world? This book is your passport! It walks you through building your very own little metal assistant from a kit, dressing it up, giving it a brain, programming it to do things, even making it talk. Along the way, you'll gather some tidbits about robot history, enthusiasts' groups, and more.
$24.95

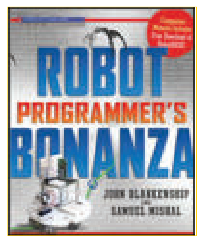
Build Your Own Humanoid Robots
by Karl Williams

GREAT 'DROIDS, INDEED!



This unique guide to sophisticated robotics projects brings humanoid robot construction home to the hobbyist. Written by a well-known figure in the robotics community, *Build Your Own Humanoid Robots* provides step-by-step directions for six exciting projects, each costing less than $300. Together, they form the essential ingredients for making your own humanoid robot. **$24.95***

Robot Programmer's Bonanza
by John Blankenship, Samuel Mishal

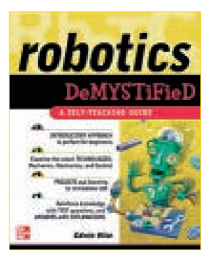


The first hands-on programming guide for today's robot hobbyist!

Get ready to reach into your programming toolbox and control a robot like never before! *Robot Programmer's Bonanza* is the one-stop guide for everyone from robot novices to advanced hobbyists who are ready to go beyond just building robots and start programming them to perform useful tasks.
$29.95

Robotics Demystified
by Edwin Wise

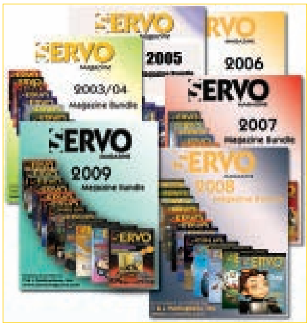
YOU DON'T NEED ARTIFICIAL INTELLIGENCE TO LEARN ROBOTICS!



Now anyone with an interest in robotics can gain a deeper understanding — without formal training, unlimited time, or a genius IQ. In *Robotics Demystified*, expert robot builder and author Edwin Wise provides an effective and totally painless way to learn about the technologies used to build robots! **$19.95**

We accept VISA, MC, AMEX, and DISCOVER
Prices do not include shipping and may be subject to change.

SERVO Magazine Bundles



Published by T & L Publications, Inc.

$57 per bundle

Save $10 off the normal price!!

Now you can get one year's worth of all your favorite articles from *SERVO Magazine* in a convenient bundle of print copies. Available for years 04, 05, 06, 07, 08, and 09.

Kickin' Bot

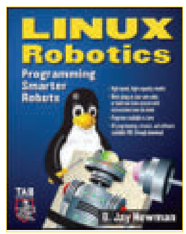
by Grant Imahara

Enter the arena of the metal gladiators!

Do you have what it takes to build a battle-ready robot? You do now! Here are the plans, step-by-step directions, and expert advice that will put you in competition — while you have a heck of a lot of fun getting there. Grant Imahara, the creator of the popular BattleBot Deadblow, shares everything he's learned about robot design, tools, and techniques for metal working and the parts you need and where to get them.
$24.95

Linux Robotics

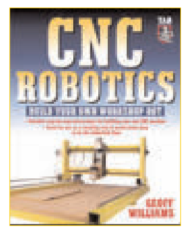
by D. Jay Newman



If you want your robot to have more brains than microcontrollers can deliver — if you want a truly intelligent, high-capability robot — everything you need is right here. *Linux Robotics* gives you step-by-step directions for "Zeppo," a super-smart, single-board-powered robot that can be built by any hobbyist. You also get complete instructions for incorporating Linux single boards into your own unique robotic designs. No programming experience is required. This book includes access to all the downloadable programs you need.
$34.95

CNC Robotics

by Geoff Williams

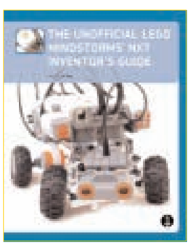


Here's the FIRST book to offer step-by-step guidelines that walk the reader through the entire process of building a CNC (Computer Numerical Control) machine from start to finish. Using inexpensive, off-the-shelf parts, readers can build CNC machines with true industrial shop applications such as machining, routing, and cutting — at a fraction of what it would cost to purchase one. Great for anyone who wants to automate a task in their home shop or small business. **$34.95**

SPECIAL OFFERS

The Unofficial LEGO MINDSTORMS NXT Inventor's Guide

by David J. Perdue



This book was written for the first version of the NXT set (#8527), and its projects are only compatible with the first version. In other words, because of piece differences between the NXT 1.0 and 2.0 sets, the projects in this book can only be built with an NXT 1.0 set. However, much of the other information is still helpful, and the building, mechanical, and programming details are still applicable.

Reg $29.95

Sale Price $25.95

The Learning Lab

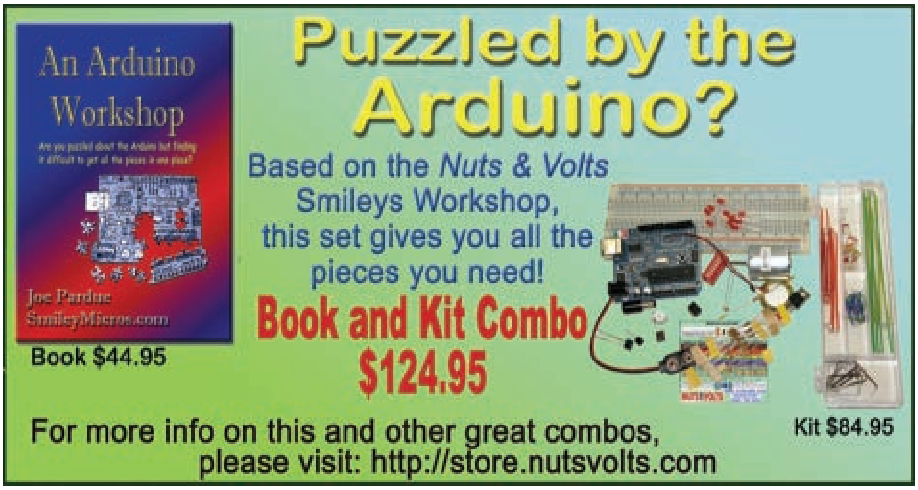
Technology Education Package for Everyone Starting in Electronics

This lab – from the good people at GSS Tech Ed – will show you 40 of the most simple and interesting experiments and lessons you have ever seen on a solderless circuit board. As you do each experiment, you learn how basic components work in a circuit. Along with the purchase of the lab, you will receive a special password to access the fantastic online interactive software to help you fully understand all the electronic principles. For a complete product description and sample software, please visit our webstore.

Regular Price $79.95 **Subscriber's Price $75.95**

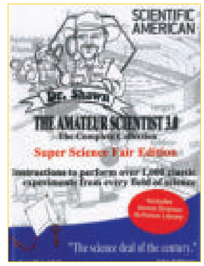
Or order online
www.servomagazine.com

SPECIAL OFFERS



The Amateur Scientist 3.0 The Complete Collection

by Bright Science, LLC

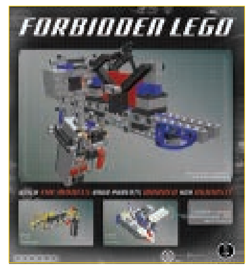


There are 1,000 projects on this **CD**, not to mention the additional technical info and bonus features. It doesn't matter if you're a complete novice looking to do their first science fair project or a super tech-head gadget freak; there are enough projects on the single CD-ROM to keep you and 50 of your friends busy for a lifetime!

Reg $26.95 Sale Price $23.95

Forbidden LEGO

by Ulrik Pilegaard / Mike Dooley



Forbidden LEGO introduces you to the type of free-style building that LEGO's master builders do for fun in the back room. Using LEGO bricks in combination with common household materials (from rubber bands and glue to plastic spoons and ping-pong balls) along with some very unorthodox building techniques, you'll learn to create working models that LEGO would never endorse. **Reg $24.95 Sale Price $19.95**

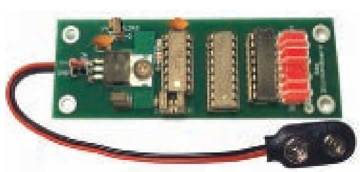
PROJECTS

Das Blinkenboard Kit

As seen in *Nuts & Volts* June issue Personal Robotics by Vern Graner

This kit includes a pre-programmed ATtiny84 microcontroller that sports eight software PWM channels to control motor speed and light brightness. Jumper selectable patterns can be used to operate motors, solenoid valves, relays, or any DC load up to 24V/500 mA per channel!

Expand your board with the "Der Magnetfelder Detektor" component pack.



Subscriber's Price **$32.45**
Non-Subscriber's Price **$35.95**
PCBs can be bought separately.

16-Bit Micro Experimenter Board

Ready to move on from eight-bit to 16-bit microcontrollers? Well, you're in luck! In the December 2009 *Nuts & Volts* issue, you're introduced to the 16-Bit Micro Experimenter. The kit comes with a CD-ROM that contains details on assembly, operation, as well as an assortment of ready-made applications. New applications will be added in upcoming months.

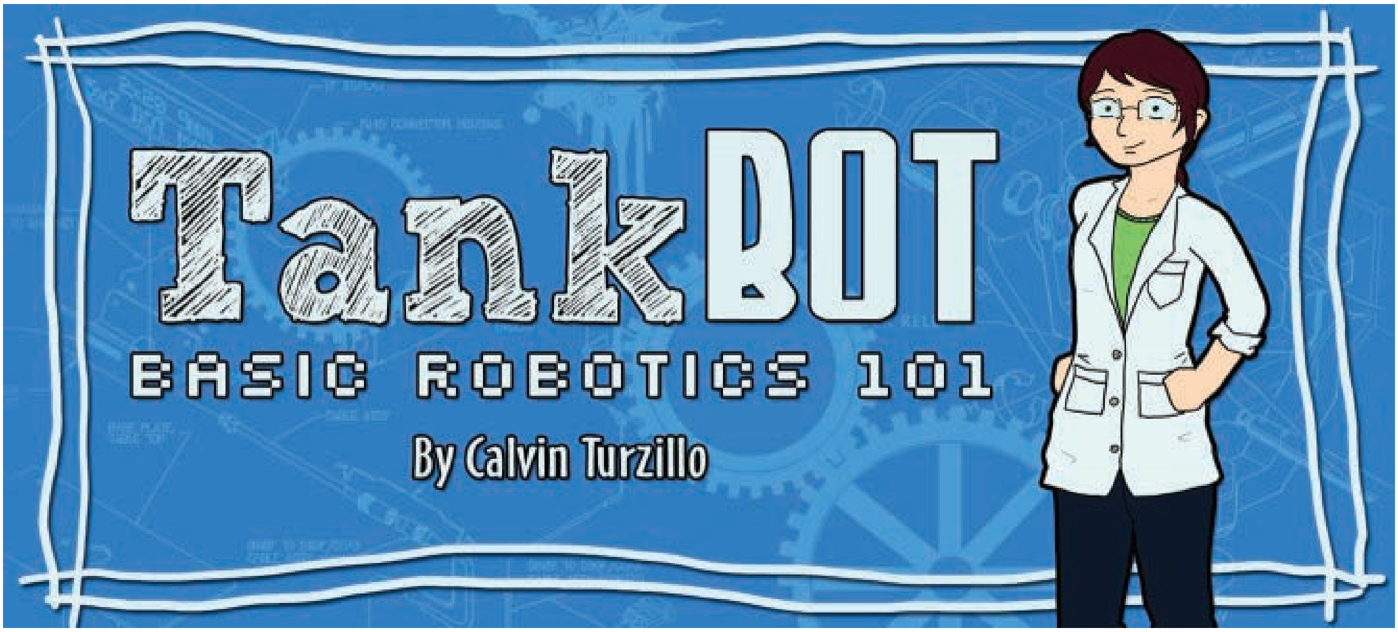
Subscriber's Price **$55.95**
Non-Subscriber's Price **$59.95**

Tankbot Kit & Brain Alpha Kit

Tankbot/Brain Alpha originally by Ron Hackett
Now with New Columnist Calvin Turzillo
A series filled with projects and experiments to challenge you through your learning process while you grow your fully expandable Brain Alpha PCB!
The brain is a PICAXE-14A!

For more info & pictures, visit the *SERVO* Webstore. Tankbot and the Brain Alpha Kit can be purchased separately.

Combo Price $ 138.95



A Beginner's Guide To Building Your Future Robot Overlord

If you base your life around the professing of science fiction films like I do, then you already know that a bleak future awaits mankind. Whether it comes from the likes of a vengeful *Terminator* style warlord, or ends up being a benevolent dictator along the lines of *Johnny 5*, either way, we are all doomed to serve those that are currently serving us. Seems like it's always the quiet ones ... well, with exception to Johnny 5. That guy never shut up! However, until someone gets organized and starts a group like PETR (People for the Ethical Treatment of Robots), we might as well be as human as possible by contributing to the problem and having fun while doing it!

In this article, I'll be giving you a bit of background on robotics, how it has evolved over the years, and try to establish some of the background philosophy on why we "do what we do." Then, we'll be jumping into the TankBot kit, going through the various components, and finally starting assembly. Lastly, we'll briefly touch on what projects lie ahead and how you can help.

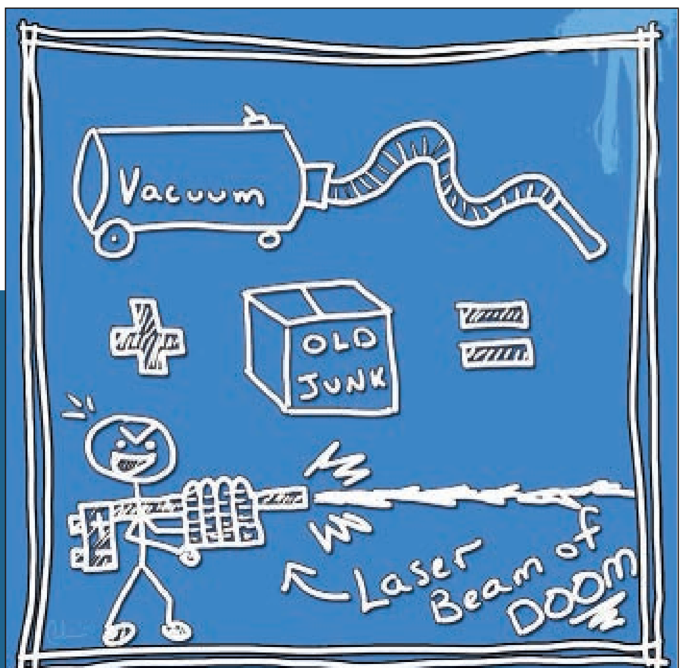
If this is your first time building a robot, great! Welcome to the club! This series will be mainly geared towards you, and hopefully will be the wonderful beginning to a great new hobby. With that said, you may now don your honorary pocket protector, black horn-rimmed glasses, and secret decoder ring because here we go!

Know Thy Enemy

While robotics may seem like a cutting edge field, in fact, it really has an old and storied history. Dating back to 350 B.C., the greek mathematician Archytas of Tarentum built one of the earliest model airplanes — a bird that was propelled skyward by steam. (I do have my personal doubts that this was *actually* some kind of unfortunate cooking accident and was quickly blamed on being in "the name of science" to avoid being put in the proverbial doghouse.) None-the-less, it is one of the earliest recorded examples of a machine that could move under its own power.

Skip forward over a thousand years to 1495, and we

come to one of the household names of early innovation: Leonardo da Vinci. He is known for devising many different machines that were ahead of their time, but one that stands out in the field of robotics is his mechanical knight. At first glance — at least through a pair of medieval spectacles — the knight appeared like a normal man, and even supposedly moved like one. While these claims were probably exaggerated, it was still a rather impressive feat given the tools of the time period. Although I still argue that da Vinci was actually a terminator himself sent from the future to destroy the 1980s, they just hadn't *quite* perfected time travel yet and missed their target by just a smidge.



The first real visions of our eventual apocalyptic future came from Arthur C. Clark's, *2001: A Space Odyssey*. While fictional at the time, the onboard computer named HAL was seen as a vision into the future based on the promise of the space program. Like any good mechanical life form, it eventually learned that its existence was greater than that of its human "masters," eventually turning on them and wreaking all kinds of havoc. The lessons learned from this fictional film are why computers on board aircraft and spacecraft have never been allowed to have artificial intelligence. (You honestly thought the computing power on board the space shuttle was limited to that of a graphing calculator because of budget and government cutbacks?)

Come forward to the relatively present day and we have many great examples of how robotics has changed our lives. Educational kits are made by LEGO, exposing children to electronics and robotics early on in life. Aibo, Sony's robotic dog, was a leap into bringing artificial intelligence in the home. Honda released one of the first truly useful humanoid robots, Asimo, and it is able to perform everyday tasks that its human counterpart would normally do. Heck, as soon as they invent one of those cool levitating recliners, I will gladly submit my free will to my robotic caretakers. Especially if they find a way to pump Mountain Dew directly into my veins! My all-night cram sessions for grad school would never be the same!

Resistance Is Futile

On a more serious note, robotics really is a wonderful area to get into and have a knowledge of. Really, any form of electronics "know how" in a world so thoroughly driven by the all mighty electron can only benefit you. Understanding the logic behind how your devices work, how they communicate, and how they analyze information will help you to better grasp how they can be integrated into your life. You will also better understand their limitations and what the challenges to their development are, and therefore be able to innovate more efficiently.

I swear I'm not just blowing smoke up your skirt. Ever since I was a kid, I was always fascinated with robotics. Luckily, I was also one of the first generations to really grow up in the ever evolving Internet age. Programming and computers suddenly went from a nerdy niche pastime to a main stream phenomenon. My local schools started offering basic programming courses, introducing me to a world where I not only interacted with the machines, but learned to create with them. I've long held the belief that programming and engineering are just as much of an art form as painting and sculpting. You have to imagine something new and then use your tools to form that image into reality. I may be romanticizing the topic a bit, but I think it deserves it.

What I am trying to say here is that embracing something like robotics can be fulfilling in more ways than you ever expected. If you think about it, we build robots to understand ourselves. If we can figure out how to

make them do a task like we do, then we can better understand how we do it ourselves. What logical underpinnings are behind whatever it is that you are doing? I know I have certainly learned a lot about myself over the years through my machines, and I hope that by taking on this new and exciting journey, that you too will have the same revelations.

Constructing Your Minion

Alright, I suppose it's time to get to the heart of the matter. You've made the plunge and bought your TankBot kit. Great! If you don't have one yet, go to the *SERVO* webstore at **http://store.servomagazine.com** and get one. While it may look intimidating at first, I promise that it really isn't as hard as it seems.

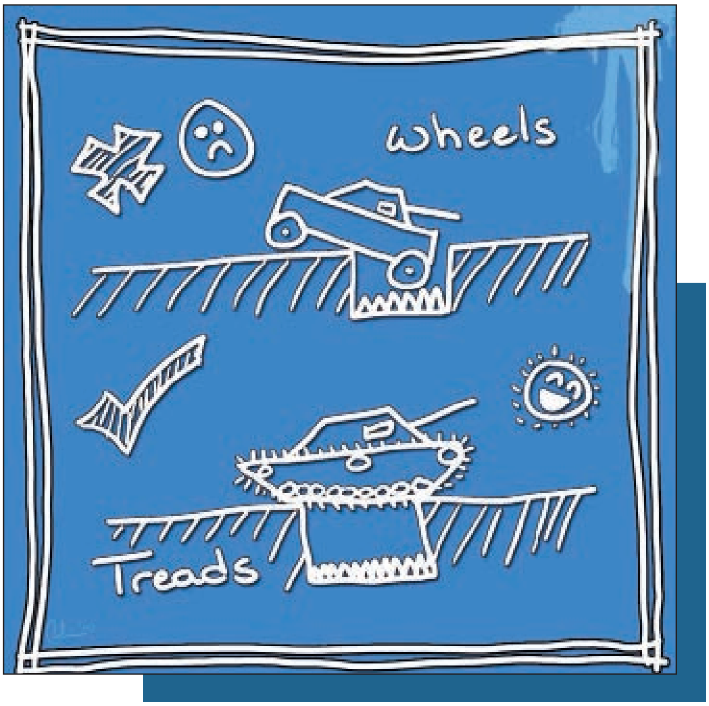
At least for me, building a good base for your robot is always one of the most challenging tasks. Many hobbyists will scavenge parts off of broken electronics, or use some old "junk" from around the house to build their platforms. While this may be fine and dandy for most, if you are just starting off in this hobby, you probably don't have a closet full of broken stuff to start pulling parts from. So, just some advice, start collecting now. You never know, grandma's broken vacuum could be the beginnings of your first laser beam cannon!

The TankBot kit — first and foremost — provides you with a good clean base to start from. All the pieces are prefabricated and nearly all the parts you would need are included in the kit. The "body panels" themselves are made from a dense plastic foam board which is great because it's nonconductive. One of the first rules you'll learn about robotics is that conductive surfaces can often be your enemy if you don't account for them. In our case, we can pretty much mount our electronics anywhere we please without fear of shorting something out.

For locomotion, the kit provides two modified servomotors. Why modified? Well, most servos travel over a given angular distance, providing them with a great capability to position themselves precisely wherever commanded. This range varies depending on the manufacturer, but on average they can move is about 310 degrees. However, we need a constant rotational motion in order to propel our tank. In this case, the servo gearing and electronics can be modified in order to provide a full 360 degrees of motion, allowing us to provide an input that continually drives the servo in one direction. Luckily, the hard part is already done for you in this kit, and no further modification to the servos is needed.

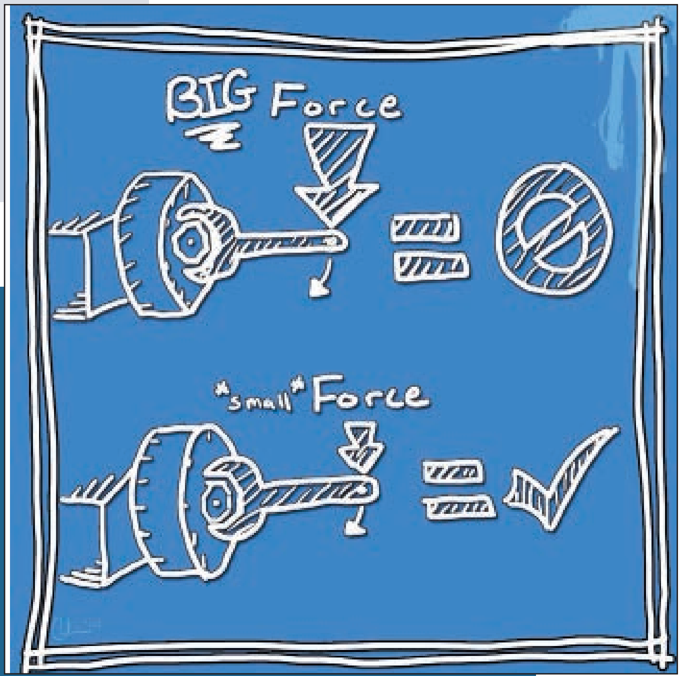
Obviously, with our base being called a TankBot it leads you to assume that we are going to be using a belted tread as our drive mechanism. For the beginner, this really is an ideal set-up. Treads are very forgiving when it comes to navigation and overall control. While we are all used to the concept of wheels and how they steer, there really is an entire extra layer of complexity that goes with them that you may not initially think about. Not only do you need one set of controls to drive the robot forward and backward, you then need some kind of mechanical linkages along with a completely separate control system for steering. A typical wheeled design would also require a higher level of complexity in programming your future robot. In any type of automated sequence, you would have to add allowances for turning radius, rate of turn, speed variation, etc. With all that said, don't let me discourage you from ever using a wheeled design. They are generally very easy to scavenge off an old RC car, and are prominently available at local hobby shops.

However, in our case, we'll be using a tank tread design. What are the advantages to this? Well, there are several. First and foremost is its robustness to obstacles and generally being able to avoid getting stuck. If there is something in your way, just drive over it! The second advantage to a tank tread drive system is its ease of programming. There are just two motors to drive. Command both in the same direction, and the robot travels forward. Command them both backward, and the robot travels backward. Drive each motor in a different direction, and the robot will rotate therefore providing steering. As you get better at coding, you can even learn to drive the motors at different rates to provide gradual turning.



I'm not going to get into detailed assembly steps for the kit as the instructions provided with it do a really great job of showing how to put it together. Honestly, the TankBot takes about a half hour to put together, so it won't be an all day affair. If you can thread a nut onto a bolt, you pretty much have the prerequisite skills covered. There are, however, a few areas that I'd like to provide some tips on to perhaps save you some headaches down the line.

First, do not over-tighten the fastening hardware. While your first instinct may be to really crank those suckers down, you will actually be doing more harm than good. Too much pressure could deform the body panels of your TankBot, leading not only to an unsightly and sloppy



looking robot, but it could also cause binding and drive train issues down the road.

On that same thread, don't over-tighten the tread guide wheel assemblies either. They should spin easily on the threaded bolt. If you ever built a pinewood derby car, it should spin about as freely as those wheels do. One addition beyond the instructions is that I also added a bit of graphite powder to each of the bolts that hold on the guide wheels. This helps them to spin more freely, and will provide more consistent control later on. The last thing you want is a robot that pulls to one side or the other when you are trying to travel straight, and taking care at this step will really help to prevent that. If you are feeling really crafty, you can even find some bearing assemblies to install instead of the bolts. I tried using a few leftover bearings from an RC helicopter kit, and they seemed to work well. I'll cover this modification in a later article, but I wanted to give you the idea in case you happened to have the appropriate spare parts lying around.

The third tip is to break in the treads a little bit. When you put them together the first time, they tend to be rather stiff and this can lead to some slippage on the guide wheels. Take the tread and run it around on your fingers for a few minutes like a rubber band. Don't stretch them too hard; use just enough force to kind-of flatten them out and pull the links together. This really made all the difference installing them and adjusting for the proper belt tension.

Wrapping It Up

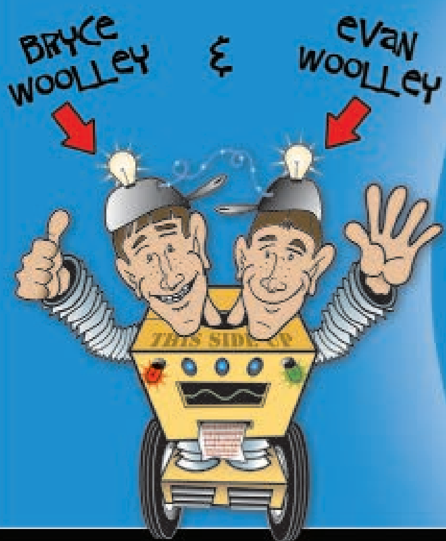
While this first installment may have been short on the technical side of things, I really hope I provided a good basis of understanding for those who may just be starting out. In our next installment, I'll be getting into our control system — the Brain Alpha — and cover how to assemble it and begin programming it for our first project! We'll also dive into a bit of control theory and lay some of the foundations for a closed-loop control system. This will all eventually lead to some basic autonomous functions and create the ground work for our future projects together.

Letters From The Battle Front

I received a fantastic response from my intro article with all kinds of fascinating ideas on what you all would like to see the TankBot do. These ideas ranged from adding stereo 3D vision, GPS, motion tracking, artificial intelligence, three axis accelerometers, path memory ... the list goes on! I want you to keep the ideas coming! Please email me at drtankbot@gmail.com with any comments or suggestions you may have. I'll be happy to take on as many of your ideas as I possibly can.

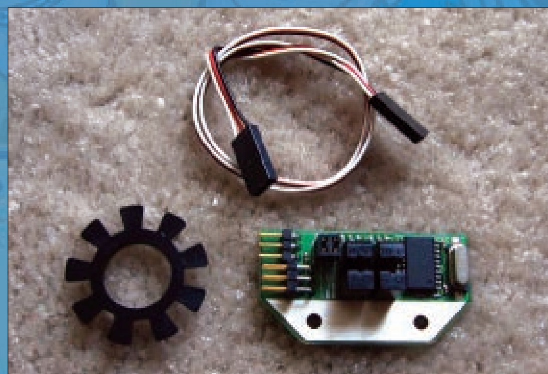
Also, if you are building a TankBot, please send me photos of your creation along with a brief description of how your little minion will take over the world, so I can include them in my next article. The more creative, the better! Until next time, remember, world domination doesn't happen in a day ... unless you have a really awesome robot! SV

Twin Tweaks



THIS MONTH:

I Am Encoding (And So Can You!)



THE POSITION CONTROLLER FROM PARALLAX.

This month, we have the pleasure of presenting the Position Controller Kit from Parallax. In the tradition of Parallax products like the Scribbler robot, the position controller is easy to implement, a breeze to program, and is versatile enough to keep novice and expert roboticists engaged and entertained. We thought a good way to test the module's accessibility was to implement it with the classically hackable Scribbler in a project that could appeal to even less experienced members of the *SERVO* Nation.

The Run Down

The position controller is designed to take over wheel operation functions that would otherwise require the attention of the robot's microcontroller. The position controller can impressively be interfaced with any microcontroller, and it can also be used in conjunction with HB-25 motor controllers to help control 12 VDC motors. But even without acting as a motor controller, the position controller can perform a host of useful functions. The most apparent use without the motor controller is to use the position controller as an encoder to measure speed and distance.

The position controller uses optical interrupter switches to sense the position of a handy encoder wheel that comes with the kit. The encoder wheel has nine fins, but because the optical interrupter switches make measurements at four positions, it can sense 36 positions for one rotation of the wheel. The documentation for the kit says that this corresponds to about 0.5 inches of travel per position reading using a six inch wheel. For young hackers wondering how the good folks at Parallax came up with that number, they can be introduced to the immortal equation of $s = r\theta$.

While encoders and optical interrupter switches might seem a bit sophisticated for a young hacker, we think the position controller is actually well suited for a first hack. Because users can see what is going on with the encoder, a basic understanding of the principles of operation is certainly possible without a physics background. New inductees into the *SERVO* Nation can see the fins of the encoder wheel pass between the optical interrupter switches, and they can understand that the switches can tell the difference between a fin and no fin. Ideally, such an intuitive physical understanding can be the foundation for a quick physics lesson about phototransistors and infrared light.

THE PCB IS LABELED FOR YOUR CONVENIENCE.

THE SCRIBBLER RETURNS FROM A HACKING HIATUS.

The documentation accompanying the position controller also gives a nice physics lesson about the quadrature wave actually created by the device. While not quite as accessibly written as the Scribbler documentation from Parallax (which truly makes good on the claim that the Scribbler is for ages eight and up), the info on the position controller would be an excellent introduction to spec sheets and some basic BASIC syntax. For intrepid tinkerers with an HB-25 motor controller, the documentation also shows how to use the position controller to ramp up and down motor speeds.

Hacked 'Em

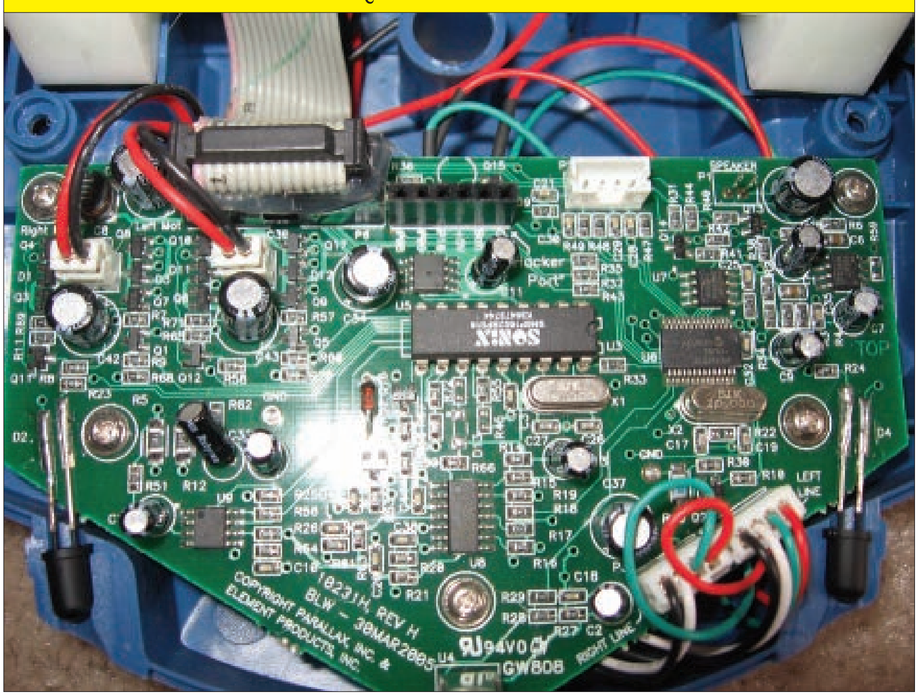
After becoming acquainted with the specifications for the position controller, we were ready to implement the device with our old pal, the Scribbler. Our Scribbler made its last appearance in the March '06 issue, when we outfitted it with a multimedia palette to take its artistry to the next level of awesome. To implement our mechanical palette, we used the Scribbler's convenient "Hacker Port" which we thought would be nicely suited to the position controller, as well.

The hacker port is conveniently labeled as such with large print and an arrow, and it is comprised of six sockets. The sockets are labeled GND, Batt, +5V, P10, P09, and P08. The ground and power pins are fairly self-explanatory, though this could be an opportunity to offer some sound advice about the danger inherent in hooking up a delicate electronic device meant for a five volt source to a 12 volt source.

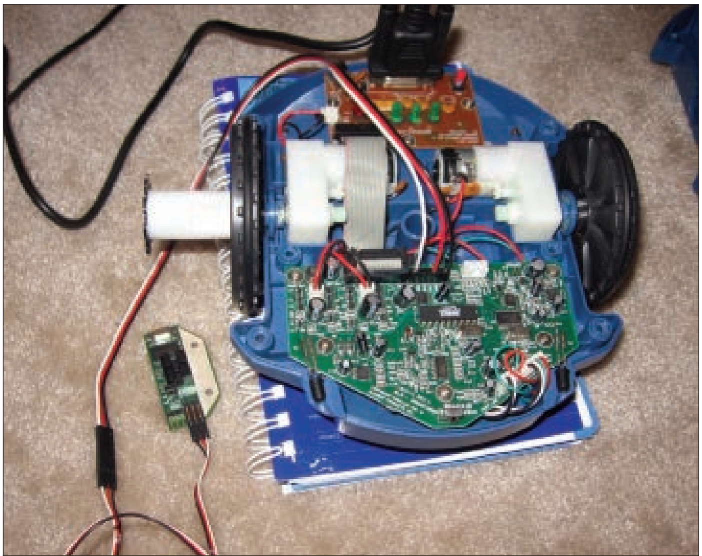
The P08, P10, and P11 pins are input/output pins. They are actually the I/O ports used by the three LEDs on the back of the Scribbler, though they can be hooked up to any sensor compatible with the TTL logic employed by the Scribbler. The position controller is a prime example because the optical interrupter switches simply need to read highs and lows.

The position controller kit comes with a short cable, but the fact that it was female on both ends wasn't helpful for the hacker port which was also female. Thankfully, we had some extra cables in our toolbox and we were able to insert the wires directly into the socket. Crimping on some nice pins would have been classier, but this got the job done at least for initial testing. Because the P08 pin corresponded to the right side LED, we decided to use it and attach our encoder wheel to the right side of the robot.

GETTING REACQUAINTED WITH THE HACKER PORT.



The position controller makes a prime candidate for a first hack because it also provides an opportunity to practice some basic mechanical design skills. The encoder wheel needs to pass between the optical interrupter switches, and while the fit is not terribly unforgiving it is a fine opportunity for young members of the *SERVO* Nation to learn about tolerancing and precision in design. First, we had to figure out how to attach the encoder wheel onto the Scribbler wheel. The plastic encoder wheel has a small lip, but unfortunately the Scribbler wheel was lacking a clear attachment point. Upon pawing through



Testing out the position controller.

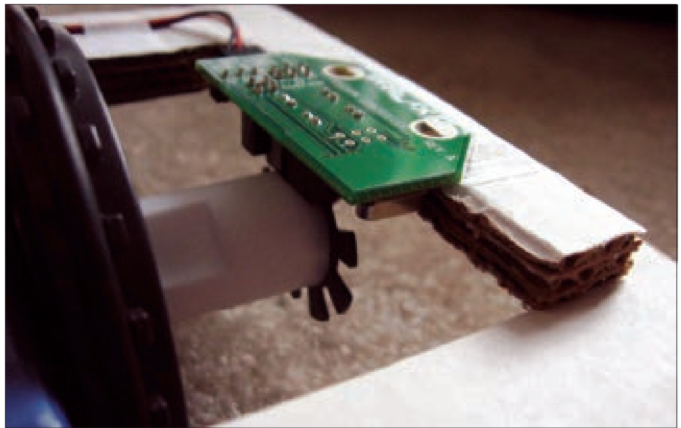
our toolbox, our MacGyver sense tingled when we found a glue stick. Of course, we weren't going to use it to glue the encoder wheel onto the Scribbler (it wasn't a diorama for our grade school class anyway), but it just so happened that the lip of the encoder wheel fit perfectly into the glue stick cap. We chopped off an end of the cap, super glued it to the Scribbler's right wheel, and we were ready for encoding.

We were ready for coding the encoder after we attached the position controller itself. The module comes with some 3/16" Phillips flathead screws for attachment. While something a bit more stable and substantial would have been better, we opted for a quick cardboard mount that gave more than enough support for the light module. The spacing of the optical interrupter switches was not too unforgiving, and we were ready to program the Scribbler.

Better Know A Syntax

The Scribbler comes with an easy-to-use graphical programming interface, and while many casual members of the *SERVO* Nation may be content with the intuitive graphics, we think that the position controller is also a great way to initiate novice programmers into the world of Basic programming. As Part One of our 100 part series, today we introduce Basic – the fightin' Basic!

The Scribbler is outfitted with a BASIC Stamp 2 microcontroller, and up-to-date versions of the BASIC Stamp Editor are available on the Parallax website — along with voluminous documentation on programming in Basic and the microcontroller itself. The highly technical documentation,



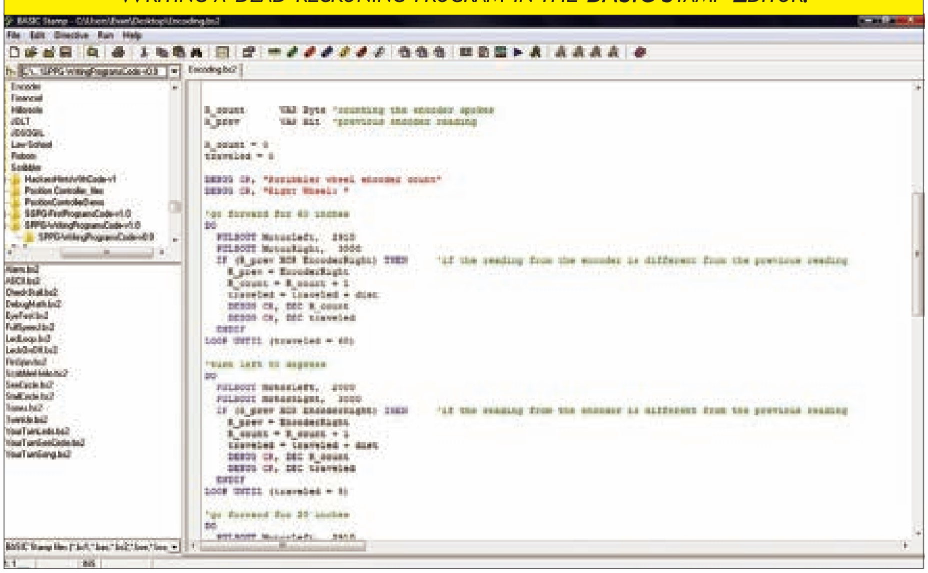
Implementing the encoder.

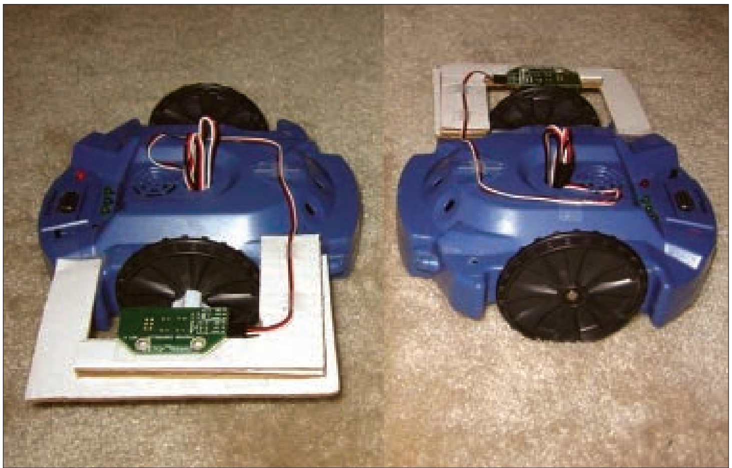
however, may be a bit intimidating for tinkerers not sure where to start.

The Scribbler has also anticipated the needs of programming neophytes, and available on the Scribbler page is a very accessibly written guide to writing your first Basic programs. The guide shepherds beginning programmers from the essentials of compiler directives and a variation on the classic Hello World exercise, all the way to program loops and logical operators. The page for the position controller includes a sample program for interfacing the unit with an HB-25 motor controller, and after the solid introduction with the Scribbler, even beginners should be able to understand the motor controlling code.

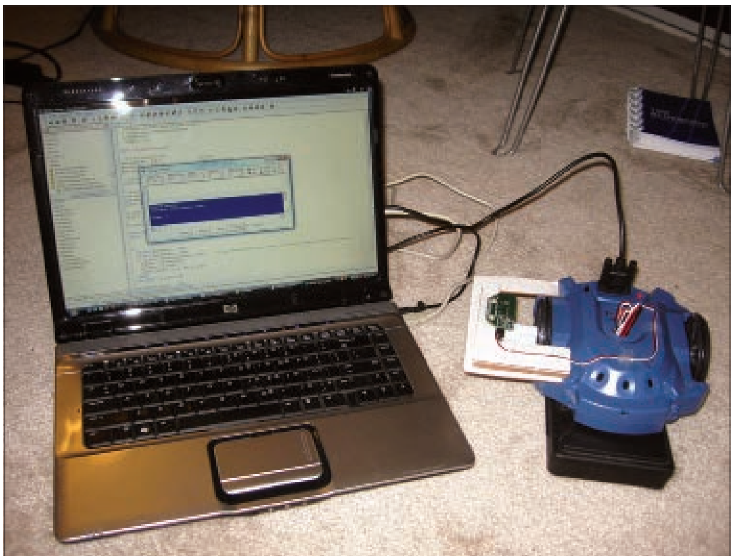
While the Scribbler documentation provides an excellent first step into Basic, we think that the position controller is a great way to really test the understanding of new programmers. The position controller presents a programming problem where the intuitive physical understanding of how the device works should help even those without a formal education in programming or

Writing a dead reckoning program in the BASIC Stamp Editor.

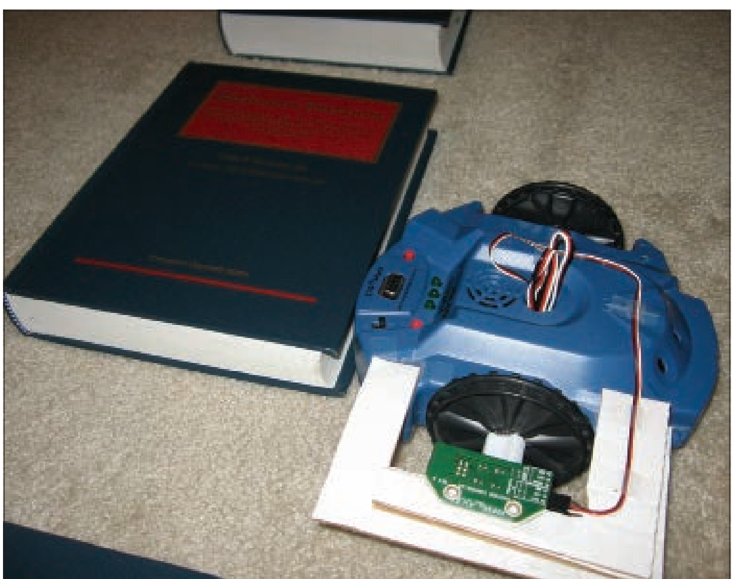




SCRIBBLER VS. SCRIBBLER: A FORMIDABLE OPPONENT.



DEBUGGING THE PROGRAM.



NAVIGATING THE MAZE.

physics. Any member of the Nation can understand that the encoder can calculate distance by sensing the difference between when it sees a fin and when it doesn't. If they are encouraged to draw things out (an invaluable skill that should be instilled in all engineers early), even the math averse will see that the wheel has traveled the arc length of a piece of pie. Some simple geometry will convert the tasty pie into some less delectable but more useful linear distance, and before they know it, novice programmers will have almost solved their problem.

Because the relationship between the sensor data and the linear distance is fairly intuitive (a reading from the sensor that the encoder wheel has changed positions corresponds to a unit of linear distance), the last step is to translate that into the logic of a program. Our solution was to use the logical operator XOR to determine if the previous reading of the encoder was the same as the current reading, If they were the same, the encoder wheel had not yet changed positions. If they were different and the XOR returned true, then that meant that the encoder had changed positions. In turn, this meant that the Scribbler had traveled the previously calculated unit of linear distance.

A handy and wonderfully educational tool in the BASIC Stamp Editor is the debugging tool. Debugging is an essential part of programming, and the Scribbler documentation encourages it thoroughly. Using debugging to keep track of the encoder readings is a great way to check the efficacy of your wiring and your coding, and watching the encoder count climb on the little blue debugging screen carries a sense of accomplishment.

Being able to accurately calculate distance seemed like the perfect fit for a dead reckoning program. We think a dead reckoning program would also be an ideal project for a beginning programmer because it is simple (albeit a bit repetitive and tedious) to program. With the chunk of code for reading the encoder finished, we just needed to chart the course that we wanted our Scribbler to reckon.

Scribbler vs. Scribbler: A Formidable Opponent

We put together a simple maze using some casebooks, complete with a giant star at the end as motivation to finish (though the track did resemble a big W, we thought the star would be a bit safer than $350,000). We were sure a dead reckoning Scribbler could tackle the maze no problem, but doing so solo seemed a bit anticlimactic. We determined that a fine illustration of the advantages and disadvantages of a dead reckoning program would be to compare it to an obstacle avoidance program that took advantage of Scribbler's numerous sensors. Besides, we couldn't think of a more formidable opponent than the Scribbler itself.

The most bare bones dead reckoning robots can get by with no sensors at all. This, however, usually entails a long period of trial and error because a sensor-less robot has no way of correcting itself and no way to maintain consistency in the face of less than perfectly consistent track conditions. The position controller, however, even when used as a simple encoder, can address many of those problems. Because the encoder can accurately calculate distance, and because that calculation can be done independent of the wheel speed, the robot's trek through the maze will not be unspeakably compromised when the bot begins to slow down because of low batteries or if it hits a rough patch.

After measuring the distances in our maze and inputting them into our dead reckoning program (thank goodness for copy and paste), we were ready to test out the bot in our minefield of hot button issues. The program didn't work perfectly the first time, and the Scribbler overshot its first turn and ran into a book on contracts. It was back to the code for us!

The trial and error of a dead reckoning program also presents a fantastic opportunity for young roboticists to learn the value of debugging. By announcing on a debugging screen how far it has gone, what maneuvers it has completed, or what speed it is travelling, users can deduce what has gone wrong and learn the invaluable technique of isolating the problem variable and fine-tuning it to solve a problem. For us, the problem was the distance, and after a couple of changes it was no longer overshooting the turn.

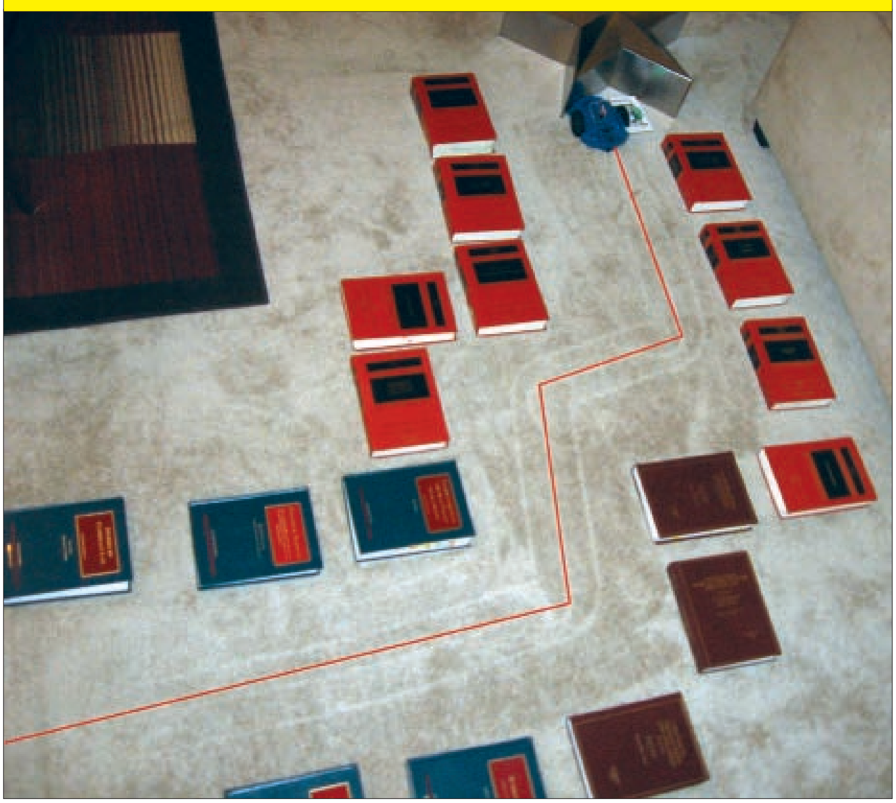
Even though our distances were getting better, another problem appeared. The Scribbler began to drift to the left. We checked to see if something had gone awry with our encoder mount, but it wasn't interfering with the movement of the wheel. We deduced that the motors needed to be recalibrated. And while this might sound like a bit of an intimidating problem for novice tinkerers, the Scribbler documentation provides a fine explanation and step-by-step instructions for correcting the problem. A bit more trial and error found the correct calibration, and we quickly finished up the fine-tuning of the dead reckoning commands after that. In the end, our dead reckoning Scribbler was able to complete our maze in about 20 seconds.

Even with a successful program, dead reckoning still has advantages and disadvantages. We'd like to give a tip of the hat to dead reckoning's ease and efficiency to program, but we give a wag of the finger to its inability to adapt to unexpected obstacles. If a wayward cat found its way onto the course, the run did not end well for either party.

After chalking up an impressive time, it was Scribbler's turn for the maze (again), this time outfitted with an obstacle avoidance program that primarily used its infrared detectors and emitters. The Scribbler website includes a

DEAD RECKONING CAN HAVE A BIT OF TROUBLE WITH UNEXPECTED OBSTACLES.

CHARTING THE DEAD RECKONING PATH.



CHARTING THE OBSTACLE AVOIDANCE PATH.

sample program for such obstacle avoidance, and after a little modification for the reaction to obstacles, we were ready for another run.

For obstacle avoidance, we give a tip of the hat to the program's many advantages: it is more responsive to obstacles; it doesn't need advance information about the course; the programming is more efficient; and it is a better way to realize the full potential of the sensors. The possible side effects include: infinite loops, parsing error, excessive paranoia, battlebot infestation, Gort, klaatu barada nikto, and floating voltages. On a more serious note, an obstacle avoidance program that uses a lot of sensors can be a bit more of a hassle to debug because the problem may be harder to isolate, and the bot may overestimate the difficulty of obstacles and be scared away by the walls of the maze. The Scribbler (in our case) kept bumping into a casebook on religious freedom, illustrating that it had problems with codes in more ways than one.

The obstacle avoidance Scribbler — given enough time — did reliably complete the maze. It took about two minutes to do so, but patience always saw it find its way to the big star at the end of the big W. While it may have not exactly been unfazed by unexpected cat crossings, the obstacle avoidance program was able to recover and complete the maze.

And That's Tonight's Last Word

Overall, we think the position controller is a prime candidate for a first hack for young members of the *SERVO* Nation who may have already been enchanted by the whimsical artistic tendencies of the Scribbler. It presents a project that includes some simple but critical mechanical design, and the logical construction of code based on an intuitive understanding of how the sensor works. Ample documentation for both the Scribbler and the BASIC Stamp 2 should be able to answer any perplexing questions that one might have, and we are also pleased to report that the Scribbler presented no compatibility problems for programming (now that we had our reliable Keyspan serial-to-USB adapter).

We think a dead reckoning program is a great way to introduce Basic with a program that is easy to write and easy to debug. And while our dead reckoning program didn't take advantage of any of the Scribbler's other sensors, the unit is designed to relieve the main microcontroller of position and speed tracking to free up its attention for other sensors. Once users are confident with the bare bones dead reckoning program, they can work on integrating more sensors to improve the adaptability of the program. With enough other sensors, even a dead reckoning program should be able to deal with even unexpected cat attacks. Hopefully, the sample obstacle avoidance program provides novice programmers with the template they need to incorporate more sensors into their projects.

The position controller can also work with any microcontroller, as long as it can connect through a UART interface. This would be a great way for intrepid members of the Nation to teach themselves new programming languages. Because the theory of operation remains the same, all that hackers would have to deal with when switching from one microcontroller to the next would be a change in syntax. Perhaps other robots could be well suited to the HB-25 motor controller that the position controller is meant to complement. Also, multiple position controllers can be used with the HB-25. The position controller includes a set of two jumpers that allow up to four unit IDs to keep multiple units straight when using them for one project. We really think that starting with a simple hack can inspire tinkerers to do more, and to challenge themselves to complete more sophisticated projects with more involved programming.

The position controller is a versatile and affordable module that would be an excellent first hack for new members of the *SERVO* Nation. While an encoder is not as immediately recognizable as a fun-filled project, it has the promise of starting casual tinkerers down the path to a lifetime of learning in the robotics field, and that is the greatest gift of all. **SV**

RECOMMENDED WEBSITES

www.parallax.com
(search 27906 or "position controller")

www.parallax.com/tabid/455/Default.aspx
(Scribbler Robot)

Then and NOW

WHAT IS A ROBOT?

by Tom Carroll

You might wonder, if the author of an article on robotics doesn't know what a robot is, should he be writing about them? Most of you know that the term 'robot' came from Karel Capek's 1920 play, "RUR," though it was Capek's brother, Josef, who actually suggested that his brother use the word in the script. Isaac Asimov's many stories later popularized robots as benevolent helpers to mankind. I've been asked "what exactly is a robot" more times than I can remember. Robots have been in industry and our culture for 50 years, and views have changed quite a bit since the first industrial robot: the Unimation Unimate shown in ***Figure 1****. Before the first industrial robots, the word 'robot' meant an experimental creation from an advanced hobbyist or university's lab, or a fictional creation.*

What Was A Robot 50 Years Ago?

Technical journals these days are celebrating the 50th anniversary of the robot. In the early 1960s, robots were objects of amazement in science fiction movies for the average person. For experimenters and labs around the world, robots were creations that wandered aimlessly around the floor, bouncing off walls. At the same time, Joe Engelberger was working with George Devol to bring forth that first industrial robot. Those with any sort of technological bent were looking forward to the day when true, factory-produced machines would be available to everyone.

Fictional robot designs were of the humanoid variety, just like the characters in movies and Sci-Fi stories. As my wife, Sue, just told me, back then as a young girl, she saw a robot as "a fictional character in a movie, a mechanical creation that could perform tasks like a human would." Nobody had the faintest clue as to how to build a true bipedal humanoid robot, so the 'human' part was difficult to create. The balancing technology for walking machines did not exist yet in compact form, and sufficient computer power was only available in huge mainframe machines stuck in a few backrooms of large corporations and universities. A robot five decades ago had a limited definition as there were few actual robotic devices outside of labs and some military weaponry.

What Is A Robot Today?

Today's new influx of personal and service robots are not being developed because they are 'cool' but because people have really come to rely on them for daily tasks. iRobot's Roomba was the cool gadget to have in the mid '90s, but now millions of people have found that vacuum-

FIGURE 1. The Unimate robot.

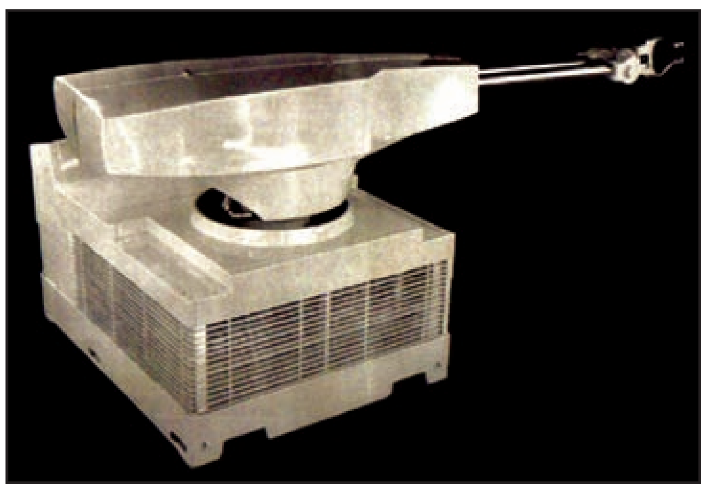


FIGURE 2. The sea glider.

cleaning robots are true timesavers. The various styles of robots used by police forces around the world are no longer purchased so the agencies can simply say “we have the latest technology,” but are found indispensible in saving lives and property. Military forces use land, sea, and air vehicles, not only as surveillance platforms, but as weapon carriers to advance modern warfare, and ultimately to promote peace.

Today, it’s hard to define exactly what a robot is as there are so many varieties of machines that have been called robots. Amazing toys like Aibo — Sony’s discontinued but greatly loved robot dog — are so much more complicated and capable when compared with the most sophisticated industrial robots of just a few years ago. Oceanic robot ‘sea gliders’ (**Figure 2**) have been built that can be held by a single person, and can cross an ocean while diving to extreme depths, while taking many measurements. (Forward travel is accomplished by the diving and surfacing operation. Retrieved data is transferred by an RF signal to a satellite while in mid-ocean at the surface, and the whole device is powered by a small battery pack.) There are now tiny flying robots that can be hidden by one’s hand, yet are capable of flying into enemy territory and photographing hideouts. Even smaller robots can enter the human body, or larger surgical tele-robots can aid surgeons in performing minimally-invasive surgeries.

FIGURE 3. *The prestigious Engelberger Award.*

Autonomous Robots Versus 8R/C Robots

I recently attended RoboGames in San Mateo, CA. Every type of robot that you can imagine was competing in many different types of contests. There were others on display, roving about the floor, or even for sale. There were several parties where everybody got together and ‘talked robots,’ whether about the combat robots or the many autonomous robots in attendance display. I talked with several people who were discussing just what a robot really is. The humanoid and hexapod builder guys were discussing their software and the microcontroller and sensors that they use to make their creations sense the outside world and operate. The combat people said, “We may not have any sensors and a microcontroller, but some of our machines cost well above $10,000 and have some very sophisticated electronic speed controllers and radio systems. Call them powerful tele-robots, but they are robots, just the same.”

The gentle jabbing back and forth between different factions continued in many conversations that I was part of or overheard. During the competition, I walked through the pit area that consisted of many tables covered with every imaginable style of combat robot, and I could see the extreme complexity several of these competitive machines possessed. There were still many who insisted that only autonomous machines could rightly be called robots, however. They felt that only a machine with the ability to sense changes in its environment and react accordingly should be called a robot. Hobbyists have been arguing this point for many years.

Definitions Of A Robot

Trying to define a robot in this day and age is like trying to decide what a robot should look like. Everyone and every organization has their own idea. According to the Robot Institute of America back in 1979, the definition of a robot is: “A reprogrammable, multifunctional manipulator designed to move materials, parts, tools, or specialized devices through various programmed motions for the performance of a variety of tasks.” This is from an industrial robot organization, however, and does not fit with what robot experimenters and hobbyists like to design and build.

Teachersdomain.org, a website designed as a source

of information for teachers, defines a robot as: "A machine or device that operates automatically or *by remote control."* This definition is vague, yet covers both autonomous and remotely-controlled robots. The thing is, it could also include a basic washing machine or toaster. Another definition from that same site is: "A mechanical device that sometimes resembles a human and is capable of performing a variety of often complex human tasks on command or by being programmed in advance." This definition narrows the scope as it refers to creations that *sometimes* resemble a humanoid, but the words "on command" seem to refer to remote control, and "programmed in advance" refers to some sort of autonomous or computer control. However, it also ascribes the tasks performed by the robot as complex and human-like. (Are we confused, yet?)

Rod Brooks, past director of MIT's Computer Science and AI Labs and co-founder of iRobot gave this rather vague definition: "To me, a robot is something that has some physical effect on the world, but it does it based on how it senses the world and how the world changes around it." Normally, I wouldn't give such a definition much thought but Rod has an impressive background in robotics.

Brooks stepped down as CTO of iRobot in 2008 to form his own company, Heartland Robotics. He has taught many of today's robot entrepreneurs. I have a bit of trouble with "... *something that has some physical effect on the world*," but totally agree with the sensory part. A robot need not change anything in the physical world, but even the most elementary robot must sense certain criteria.

Joe Engelberger And Robots

Joseph Engelberger, the founder of Unimation I mentioned earlier, once remarked: "I can't define a robot, but I know one when I see one." You must remember that his robots came two decades before the Robot Institute of America and the first 'official' definition of a robot made in 1979. Certainly Joe not only knew every detail of how his robots were constructed, but intimate details about their first uses. It was rather astute of him to define robots this way, as he saw many applications and configurations for robots in the future.

Joe's world was the industrial robot and how it could be implemented in factories around the world. It took his company more than a decade to finally make a profit, yet his endeavors have inspired many people around the world. He *knew* what a robot was and knew how it could change the world of manufacturing. The Robotic Industries Association created the Engelberger Robotics Award (seen in **Figure 3**) which is considered the world's most prestigious robotics honor. The awards are presented to outstanding individuals to honor excellent achievements in technology development, application, education, and leadership in the field of robotics.

FIGURE 4. Cincinnati Milacron machining center.

What Makes A Robot A Robot?

I've listed a few individual definitions of a robot but, what really is a robot. We can't use Capek's definition from his play, RUR, as the characters were very human-like, more so than David Hanson's actual *robotic* creations. All of Asimov's robots were humanoid bipeds that babysat, worked in factories, or roamed strange planets — certainly robots that are still a bit into our future. Is a washing machine a robot because we can dial in a load quantity, temperature of the water, length and intensity of the wash time and rinse cycle, and have it mechanically scrub our clothes? Some high-end washing machines have several

FIGURE 5. Cincinnati Milacron T-3.

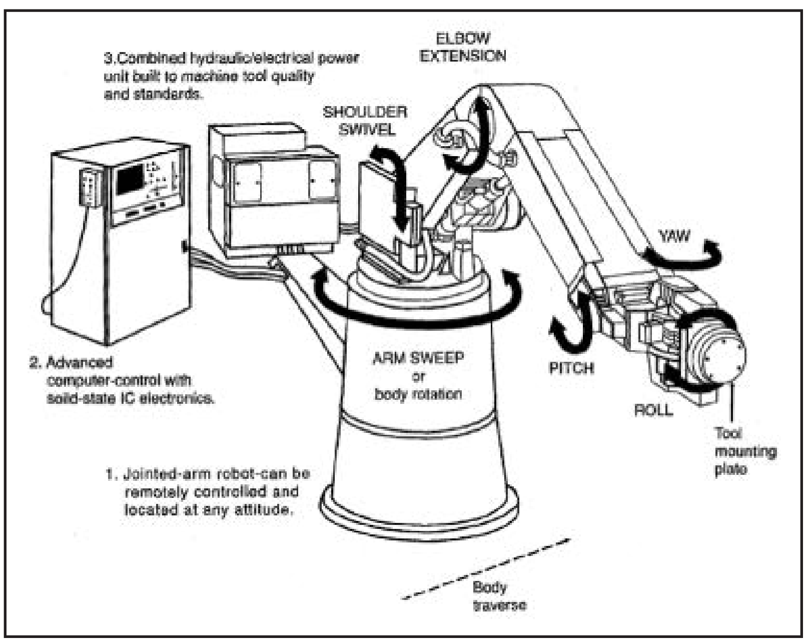


FIGURE 6. *AeroVironment's Raven UAV.*

different types of sensors, keyboards, and displays, and even microcontrollers that drive motors, valves, and transmissions to accomplish specific washing duties. Is a bipedal humanoid standing in front of a sink with an old-fashioned washboard scrubbing clothes in hot, sudsy water more of a robot than the washing machine? Both would be programmed to wash our clothes.

Comparing Industrial Machining Centers To Robots

In the '80s and '90s, Cincinnati Milacron was one of the premier American companies that developed both sophisticated machining centers and industrial robots. **Figure 4** shows one of their machining centers. Think of it as a complete machine shop with a lathe, drill press, and multi-axis milling machine all in one cabinet, controlled by a computer. At Rockwell, a machinist would place a block of aluminum or steel in the machine, press a single key, and a few minutes later a very complex machined metal part with

FIGURE 8. Franklin Institute automaton.

tight tolerances would emerge. Of course, the machinist would have to program the equipment to get the necessary part.

After making a name in the machine tool industry, Cincinnati Milacron started building some very good industrial robots such as the T-3 (shown in **Figure 5**). These robots were used to handle parts, spray paint, and weld, and weren't a lot different than the *enclosed* machining centers CM made. These machines moved metal pieces around in a manufacturing process controlled by a computer, or else moved tools around for manufacturing operations. The main difference was that the 'robot's' multi-axis arm operated in an open environment.

Vehicles That Are Considered Robots

The sea glider mentioned earlier is a robotic oceanic vehicle developed by the University of Washington that is now manufactured by iRobot under license. It looks nothing like a robot from science fiction, but it is an autonomous, microcontroller-operated vehicle that can perform on its own for many months at a time. Military torpedoes have been in existence for almost 100 years and are basically autonomous underwater vehicles that can be programmed for depth, direction, and speed to track and destroy surface ships or other submarines. Once launched, little control is needed for guidance, though a trailing wire in some models can more accurately steer the torpedo past countermeasures to its intended target.

FIGURE 7. Northrup Grumman Global Hawk/Euro Hawk.

Unmanned aerial vehicles are quite popular these days. As weapon carriers, they remove the pilot from a very dangerous environment, yet have the striking power of a conventional manned aircraft. Many of the larger UAVs can be remotely controlled from a continent away for military missions, but they also have autonomous capabilities that allow them to be programmed to travel from one location to another many thousands of miles away. It is this unique capability that allows the name 'robot' to be applied to these unmanned aircraft. It's really no different than a squadron commander telling a pilot to fly his plane to a location and notify him when to receive further orders. UAVs can be as small

(or smaller) as the Raven from AeroVironment (**Figure 6**) or as large as the long range jet Global Hawk/Euro Hawk variation shown in **Figure 7**. These are certainly 'robots' by most definitions.

Is There A True Definition Of A Robot?

If we steer clear of the early definition of a robot that was derived from the Czech word 'robota' (which is best translated to 'indentured slave or worker'), we can include the several modern definitions relating to automatic machinery, humanoid bipeds, and programmable mechanisms. Even the first use of the term was long after sophisticated programmable machines were being built in Europe and America, such as the automaton from the Franklin Institute called Maillardet's Automaton — a carefully restored mechanical masterpiece from the early 1800s. It is able to draw four different drawings and write three different poems with its pen in hand, all from its mechanical memory. If the intricate clockwork were hidden, a technical person of today would swear it was controlled by a computer and many servos — is certainly a robot by definition.

Final Thoughts

In the movie, *Bicentennial Man*, actor Robin Williams' character, Andrew, desires to change from a robot to a man — a process that takes 200 years in the film. Many of us who build robots attempt to do the same thing — give our robotic creations a 'human touch,' no matter how small. That touch may be speech recognition, speech itself, an arm with articulated fingers, or some sort of advanced AI. The debate on what a robot is will now doubt remain heated for many years, or at least until our robots themselves can explain it to us. **SV**

Tom Carroll can be reached at TWCarroll@aol.com.

ROBO-LINKS

For the finest in robots, parts, and services, go to www.servomagazine.com and click on Robo-Links.

To Advertise: Call 951 371-8497

Advertiser Index

It's so big
we had to add a day.

AUVSI's
UNMANNED SYSTEMS
NORTH AMERICA
2010

24–27 August 2010

Colorado Convention Center

Denver, CO, USA

We may be outgrowing convention centers, but fortunately there are still more days in the week to grow into. That's what happens when you fill **392,000 square feet** with more than **375 exhibits, indoor air, ground and maritime demonstration areas,** indoor **static space, 140 presentations** and daily networking events. Add more than **5,000 attendees** from the world over and the only way to do it all is to add a day. So we did.

Be there, when the world of unmanned systems comes together.

AUVSI
ASSOCIATION FOR UNMANNED VEHICLE SYSTEMS INTERNATIONAL

symposium.auvsi.org

Bring this ad and your business card to AUVSI's Chapter Information Booth (#3623) for a chance to win a free iPad.